DOI 10.37882/2223–2966.2022.07–2.14

СОЗДАНИЕ СЕРВИСОВ ПО ОЦИФРОВКЕ ШАХМАТНЫХ И БИЛЬЯРДНЫХ ПАРТИЙ

Ноек Игорь Дмитриевич
ОЧУ ВО «Еврейский университет»
noekihar@gmail.com
Замега Эмма Николаевна
К.ф.-м.н., ОЧУ ВО «Еврейский университет»
e_zamega@uni21.org
Демичев Василий Анатольевич
К.ф.-м.н., доцент, ОЧУ ВО «Еврейский университет»
vademichev@gmail.com
Блохина Ольга Анатольевна
К.т.н., доцент, Московский государственный университет технологий и управления им. К.Г. Разумовского (ПКУ)
ikafedra@yandex.ru

CREATION OF SERVICES FOR DIGITIZING CHESS AND BILLIARD GAMES

I. Noek
E. Zamega
V. Demichev
O. Blokhina

Summary. Information technologies actively absorb all branches of human activity. And sports are no exception. The digitalization of sports is currently developing in all areas: from preparing athletes for competitions to broadcasting these competitions [1]. The presented work is devoted to the development of two mobile applications for digitizing chess and billiard games for iOS and Android operating systems using artificial intelligence.

Keywords: digitalization of sports, chess, billiards, artificial intelligence, widgets, mobile applications.

Аннотация. Информационные технологии активно поглощают все отрасли человеческой жизнедеятельности. И спорт не является исключением. Цифровизация спорта в настоящее время развивается по всем направлениям: от подготовки спортсменов к соревнованиям до трансляции этих соревнований [1]. Представленная работа посвящена разработке двух мобильных приложений по оцифровке шахматных и бильярдных партий, для операционных систем iOS и Android с использованием искусственного интеллекта.

Ключевые слова: цифровизация спорта, шахматы, бильярд, искусственный интеллект, виджеты, мобильные приложения.

Введение

Стремительный рост информационных технологий коренным образом поменял мышление и привычки людей. Повсеместная цифровизация и внедрение искусственного интеллекта внесли свои коррективы и в спортивную жизнь. Спортсмены и болельщики хотят иметь доступ к онлайн-трансляциям текущих игр и к прошедшим играм. Спортивным функционерам необходимо повышать качество тренировочного процесса, а также вынесения судейских вердиктов. Все это возможно при использовании технологий по оцифровке спорта.

Информационные технологии также применяются во многих видах спорта в качестве цифрового арбитра. В теннисе, крикете, волейболе внедрён программно-аппаратный комплекс Hawk-Eye, который моделирует траекторию игрового снаряда [2]. Благодаря этой технологии повышается качество судейства турниров и зрелищность трансляций.

В футболе также осуществлялись попытки интеграции Hawk-Eye для определения забития гола, но дороговизна технологии и сложности в обслуживании привели к разработке системы видео помощи судьям VAR. Для принятия верных решений, арбитры могут согласно протоколу использования VAR обратиться к технологии и посмотреть повтор интересующего их момента с разных ракурсов и с разным темпом [3].

Иначе дело обстоит с внедрением цифровизации в такие виды спорта как шахматы и бильярд.

Цифровизация шахматных и бильярдных партий

Шахматы являются достаточно консервативным видом спорта. Интеграция информационных технологий в них происходит достаточно медленно.

В шахматах есть множество вариаций партий по времени. Для максимально достоверного и удобного обеспечения контроля времени используются специально разработанные шахматные часы.

Развитие технологий привело к созданию первых электронных часов, а в дальнейшем к созданию первых шахматных электронных моделей, обладающих рядом преимуществ: подсчёт сделанных игроками ходов, автоматический перенос времени в новый ход, поддержка различных схем подсчёта времени, остановка при просрочке времени.

Для решения проблем учёта ходов в реальном времени используется DGT электронная доска. Международная шахматная федерация использует такие доски на всех сертифицированных организацией турнирах.

Помимо громоздкой конструкции данное решение обладает высокой стоимостью, что затрудняет использование таких досок для каждой пары игроков на больших турнирах и, тем более, в небольших детских школах.

Разработанная программа для оцифровки спорта обладает преимуществами, решающими проблемы DGT доски. Данный сервис имеет мобильный форм-фактор, будучи мобильным приложением для смартфонов, сравнительно низкую стоимость, и возможность использовать Stockfish.

Stockfish — движок с открытым исходным кодом для анализа партий. Его использование позволяет после записи партии, оценить качество совершённых ходов и указать более сильные ходы по мнению алгоритма, который является 11-кратным шахматным чемпионом турнира шахматных движков. Приложение даёт пользователю возможность выбрать уровень качества анализа партии [4].

Ещё одной возможностью сервиса являются трансляции. Игроки могут в реальном времени делиться оцифрованной версией своей партии. Так же предусмотрена возможность транслировать турниры.

С точки зрения внедрения информационных технологий, бильярд является не менее консервативным видом спорта, чем шахматы.

Запись партий является главной проблемой игроков профессионалов и тех, кто начинает обучаться. Для того, чтобы демонстрировать свои лучшие удары друзьям, показывать потенциальные ошибки тренерам, игроки записывают партии на смартфоны, либо пользуются видеозаписью, организованной в клубах.

Основным функционалом разработанного сервиса является распознавание нейронной сетью ударов игрока. На данный момент приложение умеет отделять забитые удары от промахов. Эта возможность используется для нарезки партии на её лучшие моменты.

На выходе пользователь получает всю необходимую информацию о партии. Каждая игра содержит в себе метаинформацию, лучшие моменты партии в видео-формате, статистический срез об общем количестве нанесенных ударов и соотношение забитых шаров к промахам.

Выбор платформы для разработки кроссплатформенных приложений

Для успешной реализации продуктов необходимо подобрать оптимальный комплекс программно-аппаратных решений.

В качестве одного из потенциальных вариантов рассматривалось устройство в виде веб-камеры, записывающее партии и отправляющее видео-контент на сервер для последующей обработки и, при наличии устойчивого интернет — соединения, просмотра оцифрованного контента в реальном времени с помощью панели администратора.

Главным преимуществом данного подхода является стандартизированность всех используемых устройств. Это позволяет облегчить поддержку клиентов и уменьшить издержки при разработке.

Основным альтернативным кандидатом стал смартфон. Обширный выбор платформ для создания мобильных приложений, каждая из которых обладает рядом уникальных особенностей, наличие у большинства потенциальных клиентов смартфона, достаточно низкие требования для качественной работы программы, мобильные возможности устройства для передачи, просмотра, редактирования, записи стали решающим преимуществом при выборе форм-фактора устройства.

Для предоставления записи партий в условиях, не предусматривающих стабильное интернет — соединение принято решение интегрировать нейронную сеть в само приложение. Данный подход позволяет производить запись видеопотока, например, в любом бильярдном клубе, многие из которых расположены в подвальных помещениях со слабым уровнем сигнала.

Анализ преимуществ и недостатков, приведённых выше решений, выявил, что смартфон является наиболее выигрышным вариантом.

Кроссплатформенная разработка открывает возможность создания приложения сразу для нескольких операционных систем на единой кодовой базе.

Крупные корпорации сами создают и развивают кроссплатформенные фреймворки: Google — Flutter, JetBrains — Kotlin Mobile Multiplatform, Microsoft — Xamarin, Facebook — React Native.

На момент планирования и начальной стадии разработки на рынке представлено два стабильных, развивающихся и функциональных решения для создания кроссплатформенных приложений, описанных ниже [5].

React Native — фреймворк с открытым исходным кодом для разработки мобильных приложений на языках программирования Java Script и Type Script.

Исполняемый код программы взаимодействует с Application Programming Interface нативных платформ посредством инструмента Bridge.

Одним из главных преимуществ React Native является низкий порог входа в него для веб-разработчиков. Предполагается, что им будет достаточно легко начать создавать мобильные приложения для Android и iOS [6].

На момент выполнения работы, популярность Flutter растёт значительно быстрее ReactNative. Это происходит за счёт следующих заметных преимуществ фреймворка от Google: более высокий уровень производительности приложений и превосходящие по удобству инструменты разработки, позволяющие ускорить и упростить создание приложений, что приводит к более быстрому развитию платформы и выбору компаний [4].

Для осуществления быстрой и стабильной разработки мобильных сервисов, прежде всего, необходимо подобрать подходящие инструменты разработки [7]. В частности, архитектуру, state-manager, плагины для локализации приложения, генерации моделей с данными, REST api сервисов и так далее.

За счёт декларативного подхода к построению пользовательского интерфейса, фреймворк Flutter позволяет быстро создавать гибкий, модульный дизайн приложения. Однако это приводит к издержкам при передаче данных между различными view.

В большом проекте данная проблема будет наиболее проблематична по двум причинам: данный код будет являться копией себя и в случае внесения правок возникнет необходимость в редактировании данной логики во всех участках кода, где она используется.

Помимо потребности в обработке данных в разных участках кода есть дополнительные потребности в использовании State Manager в приложении –это разделение UI с бизнес логикой и имплементация принципов качественной разработки крупных сервисов.

Flutter предлагает два основных типа виджетов — Stateless и Stateful [8]. Первый тип не обладает состоянием и не может обновляться без внешних событий, возникающих на родительских виджетах — контейнерах.

Виджеты способны обновлять интерфейс во время выполнения программы. Для того, чтобы указать на то, что состояние изменилось, необходимо вызывать метод setState, который является самым быстрым и простым решением проблемы. Но он далеко не всегда позволяет инкапсулировать бизнес-логику от пользовательского интерфейса. Решением данной проблемы служат более мощные и функциональные решения. На данный момент, чаще и успешнее всего используются provider, bloc/cubit, redux/fish_redux.

Provider является самым простым из приведённых выше решений, позволяющим в кратчайшие сроки решить проблему state management, но и это приводит к ряду архитектурных ограничений, которые не всегда способны помочь правильно решить проблему, особенно, в большом приложении.

Хуже всего provider подходит для глобальных состояний приложения, например, авторизации. Лучшим решением для данного сценария является redux, который пришел во Flutter из веб-разработки.

К преимуществам redux относятся:

- контроль над состоянием — всегда известно, чем вызвано в нем изменение;
- функции обновления состояния легко тестировать;
- приложение легко поддаётся разбиению на различные слои;
- стейт глобальный и нет необходимости каждый раз передавать его в различные участки приложения.

За счёт этих преимуществ redux можно назвать крайне качественным и функциональным решением для больших приложений. Redux отлично подходит для реализации сложных сценариев, в которых отдельные участки логики должны реагировать на действие. Архитектура redux приведена на рисунке 1.

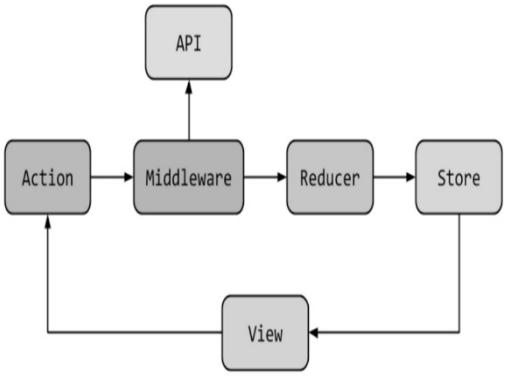

Рис. 1. Архитектура redux

Данная схема работает по следующему принципу: после срабатывания Action — некого действия из интерфейса приложения оно отправляется в middleware для обращения к data слою, далее обновляется состояние и возвращается во view.

Redux — одно из самых гибких функциональных решений, но при этом, он, за частую излишне массивен и сложен в обслуживании.

Bloc — так же, как и redux способен решать задачи повышенной сложности. Но при этом он гораздо более простой во внедрении и поддержке.

В определённое место дерева виджетов добавляется один из видов виджетов bloc — для обработки состояния, который, при каждом обновлении состояния будет обновлять дочерние элементы.

Cubit является облегченной версией bloc, главным отличием которой является то, что на вход cubit получает функции, а не события. Это приводит к тому, что состояния будут возвращаться в порядке запросов событий [9].

Во время разработки сервисов использовались все три state manager, но итоговым вариантов стал cubit. На данный момент он сочетает в себе простоту в обращении и предоставление функционального интерфейса, который позволяет решать любые задачи.

Для сбора аналитики, загрузки удалённой конфигурации, мониторинга ошибок, обработки push-уведомлений используются сервисы Firebase.

Firebase — многофункциональная платформа, разрабатываемая и поддерживаемая Google — той же компанией, что и Flutter. Благодаря этому интеграция, поддержка и разработка сервисов данной платформы в приложения является крайне простой.

Firebase crashlitics предоставляет возможность собирать информацию о произошедших у пользователей ошибках [10].

Данный сервис незаменим для любого крупного приложения, так как он позволяет оперативно находить ошибки, упущенные в процессе тестирования и исправлять их.

Отправка push-уведомлений для каждой мобильной платформы осуществляется посредством нативного api. Flutter — нативный фреймворк, поэтому всё, что приводит к персонифицированной разработке для каждой платформы является большой издержкой. Firebase для Flutter предоставляет универсальный интерфейс для отправки уведомлений.

Firebase cloud messaging позволяет отправлять push-уведомления в приложения с поддержкой дополнительного функционала [11].

В данной задаче важно использовать максимально удобные и гибкие инструменты по причинам снижения издержек при разработке компонентов пользовательского интерфейса и возможности оперативного редактирования локализации.

В качестве инструментов локализации использовалась связка удалённой конфигурации Firebase и плагина easy_localization. Первый сервис позволяет без обновления приложения редактировать локализацию. Несмотря на то, что этот способ является предпочтительным, важно проработать сценарий, в котором получить данные через интернет не удастся. Плагин easy_localization предоставляет набор удобных инструментов для локализации приложения без доступа к сети.

Для удобства интеграции локализации в пользовательский интерфейс написана вспомогательная функ-

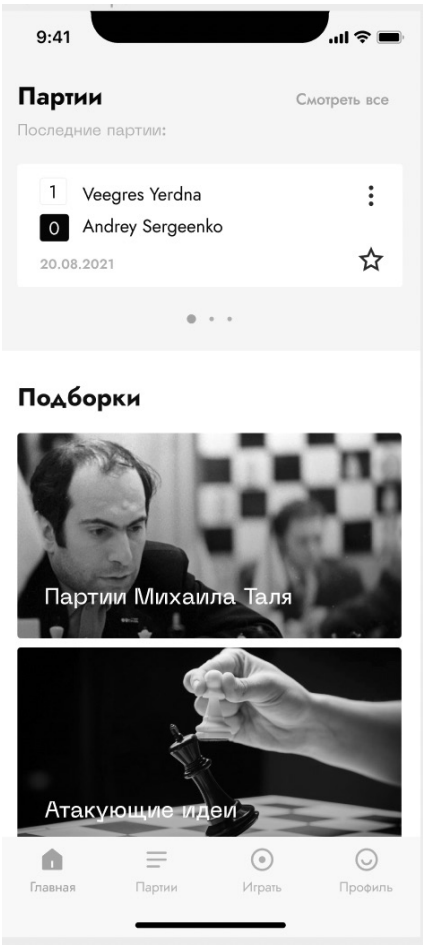

Рис. 2. Пример главной страницы

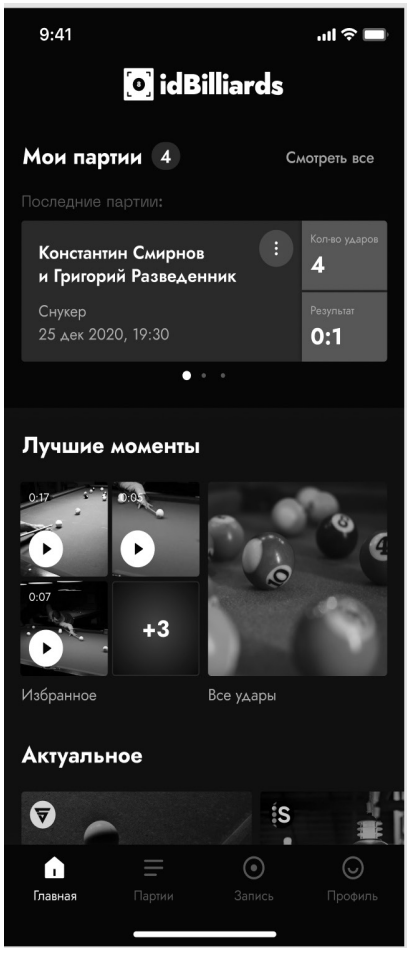

Рис. 3. Главная страница приложения для бильярда

ция, в которую внутри ui достаточно передать ключ из json файла.

Flutter предлагает два api для осуществления навигации в приложениях, которые принято называть Navigator и Navigator 2.0.

Первая версия полностью императивная. Она содержит в себе объект Navigator, который управляет PageRoute, являющимися привычными окнами в приложении или модальными окнами [12].

Navigator во всех случаях работает по принципу стека. Экран вверху стека видит пользователь.

Преимуществом первого подхода является простота в применении, а недостатком — слабые возможности. Второй подход позволяет реализовывать навигацию разной сложности, но обладает сложным интерфейсом взаимодействия. В качестве Navigator разработанных сервисов используется плагин auto_route [13].

В приложении разработаны четыре секции: главная страница, список оцифрованных партий, запись партии и создание трансляций, профиль.

Главная страница (рисунок 2) представляет собой оцифрованную подборку интересных шахматных партий.

С их помощью можно изучать игры профессионалов, подмечать интересные ходы в рамках специально разработанного шахматного плеера, который предоставляет возможность ставить игру на паузу, ускорять или замедлять её, переворачивать доску, выбирать нужный ход и переключаться на состояние доски, актуальное на момент хода.

Когда у пользователя появятся собственные оцифрованные партии, на главной странице появится список из карточек с метаинформацией о последних трёх партиях. В данной секции можно редактировать данные о партии, добавлять её в избранное, делиться оциф-

рованной партией в форматах GIF и PGN. С этого экрана можно так же попасть в список всех оцифрованных пользователем игр.

Страница с партиями может быть в двух глобальных состояниях, когда она содержит оцифрованные партии, или сохранённые партии отсутствуют.

В первом состоянии на данном экране находится список карточек со всеми оцифрованными играми пользователя. Карточки обладают идентичным функционалом аналога с главной страницы, описанного выше.

Страница с бильярдными партиями может быть в таких же двух глобальных состояниях, как и в приложении для шахмат.

Ниже плеера расположена секция с анализом партии. Анализ производится посредством движка Stokfish.

На экране с записью партий пользователь может начать оцифровку своей партии, создание трансляции своей партии и подключение к уже созданному турниру.

При начальной настройке записи партии есть возможность заранее ввести данные о партии: имя игрока за белых; имя игрока за чёрных; название турнира; место проведения турнира; дата. Если пользователь решит создать трансляцию своей партии, то сможет сразу поделиться ссылкой на неё в социальных сетях. Страница профиль содержит информативные разделы, такие как информация о приложении, пользовательское соглашение.

Помимо ознакомительной информации в профиле присутствуют разделы с информацией о текущей подписке пользователя, его личных данных, которые можно редактировать и настройки приложения. Последние позволяют редактировать push-уведомления, например, выбрать время, в которое приложение напомнит пользователю потренироваться.

В разделе настройки присутствует возможность выбора параметров анализа партий.

Одной из самых важных отличительных особенностей приложения является возможность создавать турнирные трансляции. Администратор создаёт турнир в специально разработанном клиенте. Данный подход открывает доступ к ряду преимуществ. Подключиться к созданной трансляции можно через веб-браузер с любой платформы. Использование единой платформы для разработки мобильного приложения позволяет упростить взаимодействие между этими клиентами.

Приложение для бильярда

В шахматах, для сохранения партий активно используется формат PGN — portable game notation. Он отлично подходит для решения задач по оцифровке игр. Благодаря распространённости и популярности, он применим ко множеству шахматных сервисов.

В бильярде, на данный момент не существует аналогичных форматов партий. Для разработанного приложения, после консультаций с функционерами в области бильярда, принято решение, после оцифровки игры предоставлять пользователю видеофрагменты с записанным звуком.

При разработке приложений сделан упор на создание модульных компонентов в дизайне, мобильной разработке и её серверной части. Это позволяет создать единый узнаваемый дизайн-стиль продуктов компании для разработки спорта.

Главная страница (рисунок 3) помимо незначительных отличий в дизайне, содержит набор актуальных новостей из мира бильярда, предварительный просмотр партий и лучших моментов пользователя.

Остальные экраны имеют аналогичный основной функционал.

Функционала трансляций нет в данном приложении, так как в большинстве бильярдных клубов отсутствует устойчивое интернет — соединение. В дальнейшем, если будет высокий спрос на данный функционал, будет рассмотрена возможность её интеграции в приложение.

Для оптимизации работы мобильного устройства во время продолжительной тяжелой нагрузки — записи видео, приложение предоставляет возможность выбрать качество записи.

Заключение

Разработанные сервисы по оцифровке шахматных и бильярдных партий, посредством нейронной сети, переводят реальные игры в цифровой формат данных, который позволяет производить интеграции со сторонними сервисами и создавать собственные. В частности, в удобном формате сохранять любимые игры, делать статистические срезы в удобном формате и с максимальной точностью определять нарушения правил.

Приложение для оцифровки бильярда обладает схожей концепцией, но акцент в нём делается на лучшие удары. По завершении каждой партии пользователь получает нарезку с самыми яркими ударами, информацией

о партии и итоговому статистическому срезу о данной игре.

Созданные приложения idChess и idBilliards позволяют производить дальнейшую обработку оцифрованных партий, idChess и idBilliards прошли апробацию и активно распространяются.

На данный момент приложение idChess уже можно скачать в магазинах приложений AppStore и PlayMarket [14]. На данный момент множество турниров различных масштабов по всей стране оснащены idChess. С каждым днём у него появляются новые пользователи, которые оставляют положительную обратную связь, а турниры в различных странах мира внедряют приложение.

Одной из ключевых будущих особенностей шахматного сервиса станет возможность определять невозможные ходы, что станет революцией в проведении турниров.

Эти сервисы положили начало линейке приложений idSport компании FriFlex. В дальнейшем, на базе разработанных платформ будут разработаны приложения для других спортивных дисциплин.

ЛИТЕРАТУРА

1. Использование информационных технологий в спорте// [электронный ресурс]: https://expeducation.ru/ru/article/view?id=5597
2. Hawk-Eye //: https://ru.wikipedia.org/wiki/Hawk-Eye.
3. Video Assistant Referee // [электронный ресурс]: https: //www.fifa.com/technical/football-technology/standards/video-assistant-referee
4. Flutter Apps // [электронный ресурс]: https: //www.intelivita.com/blog/apps-made-with-flutter/
5. Сравнение Flutter и React Native// [электронный ресурс]: https: //surf.ru/flutter-vs-react-native-chto-vybrat-dlya-vashego-mobilnogo-prilozheniya/
6. React Native Apps// [электронный ресурс]: https://fireup.pro/blog/9-amazing-mobile-apps-built-with-react-native.
7. Ноек И.Д. Разработка мобильного приложения с использованием Framework Realm // Материалы Ежегодной межвузовской студенческой научной конференции ОЧУ ВО «Еврейский университет». Сборник тезисов. Москва, 2020. С. 324–331
8. История развития шахматных часов// [электронный ресурс]: https: //ru.wikipedia.org/wiki/Шахматные_часы.
9. DGT Board // https:// www.chessprogramming.org/DGT_Board.
10. Firebase crashlytics// https:// firebase.google.com/docs/crashlytics.
11. Firebase cloud messaging// [электронный ресурс]: https:// firebase.google.com/docs/cloud-messaging.
12. Navigator// https:// api.flutter.dev/flutter/widgets/Navigator-class.html.
13. Flutter auto_route plugin// https:// pub.dev/packages/auto_route.
14. idChess AppStore// [электронный ресурс]: https:// apps.apple.com/ru/app/idchess-play-and-learn-chess/id1464126978.