ООО «МИ-МИ»
Правообладатель программного обеспечения «FinMe»

Описание жизненного цикла программного обеспечения «FinMe»

Документ подготовлен для включения сведений о программном обеспечении в единый реестр российских программ для электронных вычислительных машин и баз данных

Москва, 2026

1. Общие сведения о программном обеспечении

Настоящий документ описывает жизненный цикл программного обеспечения **«FinMe»** (далее — программное обеспечение, программа, продукт). Документ подготовлен правообладателем и предназначен для представления в уполномоченный орган при включении сведений о программе в единый реестр российских программ для электронных вычислительных машин и баз данных.

Правообладателем программного обеспечения является **ООО «МИ-МИ»** (далее — организация, правообладатель). Организация обладает исключительным правом на программу в полном объеме и осуществляет ее дальнейшее сопровождение, доработку и распространение.

Программное обеспечение «FinMe» представляет собой **универсальную систему управления партнерскими сетями по модели оплаты за целевое действие (CPA — Cost Per Action, оплата за целевое действие)**. Продукт предназначен для организации взаимодействия между участниками партнерской сети, учета и обработки целевых действий, распределения трафика между партнерскими предложениями, а также для сбора и представления отчетности о результатах работы сети.

По типу реализации программа является **клиент-серверным веб-приложением**. Модель предоставления — **SaaS** (программное обеспечение как услуга): доступ к продукту осуществляется через веб-браузер, установка на устройство пользователя не требуется. Пользователь работает с программой удаленно, используя стандартные средства доступа к сети.

Программное обеспечение построено на следующих технологиях:

- **Серверная часть** — язык программирования Java 21, платформа Spring Boot 3.3.4.
- **Клиентская часть** — язык программирования JavaScript, платформа Vue.js.
- **Базы данных** — PostgreSQL 15 для хранения оперативных данных и ClickHouse для аналитического хранилища отчетов.
- **Объектное хранилище** — S3 для хранения файлов и вспомогательных данных.

Важной особенностью жизненного цикла продукта является порядок его приобретения организацией. Разработка программного обеспечения «FinMe» была **завершена до учреждения ООО «МИ-МИ»**. Организация получила программу в готовом виде и является правообладателем готового продукта. С момента получения прав организация осуществляет сопровождение, поддержку и доработку программы. Выпуск обновлений выполняется вручную по внутреннему регламенту организации.

2. Модель жизненного цикла

Жизненный цикл программного обеспечения «FinMe» организован в соответствии с положениями национального стандарта **ГОСТ Р ИСО/МЭК 12207 «Информационная технология. Системная и программная инженерия. Процессы жизненного цикла программных средств»**. Стандарт определяет набор процессов, действий и задач, применяемых на протяжении жизни программного средства.

К программному обеспечению «FinMe» применяется **эволюционная (итеративная) модель поддержки готового продукта**. Выбор данной модели обусловлен тем, что программа была получена организацией в завершенном виде, а дальнейшая работа над ней носит характер постоянного сопровождения, доработок и регулярного выпуска обновлений. Продукт развивается небольшими последовательными улучшениями, каждое из которых проходит полный цикл от постановки задачи до выпуска и передачи в эксплуатацию.

Эволюционная модель позволяет организации гибко реагировать на потребности пользователей и изменения условий работы, постепенно наращивать возможности программы и своевременно устранять выявленные недостатки, сохраняя при этом стабильность работающего продукта.

Жизненный цикл программного обеспечения включает следующие стадии:

1. Формирование требований.
2. Разработка.
3. Тестирование и контроль качества.
4. Выпуск и развертывание.
5. Эксплуатация и поддержка.
6. Вывод из эксплуатации.

Стадии формирования требований, разработки, тестирования, выпуска и эксплуатации выполняются циклически: по мере накопления задач и обращений пользователей организация проходит их повторно, выпуская новую версию продукта. Стадия вывода из эксплуатации выполняется однократно при прекращении использования программы.

3. Стадия формирования требований

На стадии формирования требований определяется перечень доработок и изменений, которые должны быть внесены в программное обеспечение. Требования формируются на основании нескольких источников:

- Обращения пользователей, поступающие по каналам поддержки.
- Внутренние решения организации о развитии продукта и улучшении его возможностей.
- Результаты анализа работы программы и выявленные в процессе эксплуатации недостатки.

Учет и систематизация требований выполняются в системе управления задачами **Яндекс Трекер**. Каждая задача на доработку, исправление или улучшение оформляется в виде отдельной записи, для которой указывается описание, приоритет и ответственный сотрудник. Дополнительно сотрудники используют личные канбан-доски для планирования и отслеживания хода выполнения работ.

Порядок фиксации требований обеспечивает прозрачность и управляемость процесса: все поступившие задачи хранятся в едином реестре Яндекс Трекера, что позволяет контролировать их состояние, определять очередность выполнения и планировать состав очередного обновления. По накоплению достаточного числа проработанных задач принимается решение о переходе к следующим стадиям жизненного цикла.

4. Стадия разработки

На стадии разработки выполняется внесение изменений в исходный код программного обеспечения в соответствии с задачами, поставленными на предыдущей стадии. Работа с исходным кодом ведется в приватных репозиториях системы контроля версий **Git**.

Хранение кода организовано на **самостоятельно управляемом сервере GitLab**, размещенном на инфраструктуре Yandex Cloud на территории Российской Федерации. Использование собственного сервера контроля версий обеспечивает сохранность исходного кода и полный контроль организации над средствами разработки.

Организация применяет следующую модель работы с ветками репозитория:

- **Основная ветка** используется для хранения стабильного состояния продукта, готового к выпуску.
- **Отдельные ветки** создаются для выполнения конкретных доработок и исправлений; работа над задачей ведется изолированно, не затрагивая стабильное состояние.
- После завершения работы над задачей изменения из отдельной ветки объединяются с основной веткой.

Перед объединением изменений выполняется внутренний просмотр кода ответственным сотрудником: проверяется соответствие внесенных изменений поставленной задаче, корректность и качество кода. Только после успешного просмотра изменения объединяются с основной веткой.

Среда разработки включает средства работы с указанными технологиями (Java 21, Spring Boot, JavaScript, Vue.js) и локальные средства запуска компонентов программы. Все изменения отражаются в истории репозитория, что обеспечивает прослеживаемость внесенных доработок и возможность возврата к предыдущим состояниям кода.

5. Стадия тестирования и контроля качества

Перед выпуском обновления программное обеспечение проходит проверку качества. Тестирование выполняется по внутреннему регламенту организации и включает несколько видов проверок, дополняющих друг друга.

Применяются следующие виды проверок:

- **Ручное тестирование** по проверочным сценариям основных функций продукта: ответственный сотрудник последовательно проверяет работу ключевых возможностей программы и убеждается в их корректной работе.
- **Проверочные (smoke) проверки** после сборки: выполняется быстрая проверка работоспособности собранной версии для подтверждения того, что основные функции запускаются и работают.
- **Автоматические модульные тесты** для отдельных подсистем: часть логики покрыта автоматическими тестами, которые запускаются при внесении изменений.
- **Проверка типов и статический анализ кода** для клиентской части: выполняется автоматическая проверка соответствия типов и анализ кода на предмет возможных ошибок.

Сочетание ручных и автоматических проверок позволяет выявлять недостатки до передачи обновления пользователям и обеспечивать стабильность работы продукта. При обнаружении ошибок задача возвращается на стадию разработки для исправления, после чего проверка повторяется. Обновление допускается к выпуску только после успешного прохождения всех предусмотренных проверок.

6. Стадия выпуска и развертывания

После успешного прохождения проверок качества обновление переходит на стадию выпуска и развертывания. На этой стадии проверенная версия продукта готовится к установке и вводится в работу.

Выпуск и развертывание выполняются в следующем порядке:

7. Выполняется сборка компонентов программы в образы контейнеров с использованием технологии контейнеризации **Docker**.
8. Готовые образы публикуются в реестр контейнеров **Yandex Container Registry**.
9. Развертывание образов выполняется в среде **Yandex Managed Service for Kubernetes**.
10. Фиксируется значение версии выпущенного обновления.

Фиксация версий выполняется по следующей схеме. В системе контроля версий на состояние выпущенной версии устанавливается **тег вида vX.Y.Z**. Значение версии дополнительно дублируется в файле сборки **Makefile**. Перечень внесенных изменений ведется в истории репозитория, что позволяет однозначно сопоставить каждый выпуск с составом изменений.

Выпуск и развертывание носят **ручной характер**: все действия выполняются ответственным сотрудником по внутреннему регламенту. Решение о выпуске новой версии принимается по накоплению задач в Яндекс Трекере, по обращениям пользователей и по внутренним решениям о развитии продукта. Ручной порядок выпуска обеспечивает контроль над каждым обновлением и снижает вероятность ошибок при вводе версии в работу.

7. Стадия эксплуатации и поддержки

После развертывания программное обеспечение переходит в стадию эксплуатации. На этой стадии пользователи работают с продуктом, а организация обеспечивает его поддержку, устраняет выявленные недостатки и отвечает на обращения.

Поддержка пользователей и обработка инцидентов организованы следующим образом:

- Обращения пользователей принимаются через **мессенджер** и **электронную почту**.
- Для обработки обращений и инцидентов назначен **ответственный разработчик**.
- Предусмотрены **автоматические уведомления об ошибках уровня ERROR**, которые направляются в мессенджер и позволяют оперативно узнавать о возникающих сбоях.
- Исправления выявленных недостатков выпускаются в составе обновлений продукта.
- Целевой срок первичной реакции на обращение — **в течение рабочего дня**.

Такая организация поддержки обеспечивает своевременное реагирование как на обращения пользователей, так и на автоматически выявленные ошибки. Поступившие в ходе

эксплуатации задачи учитываются в Яндекс Трекере и при необходимости включаются в состав очередного обновления, что замыкает жизненный цикл продукта на новую итерацию сопровождения и доработки.

8. Стадия вывода из эксплуатации

Стадия вывода из эксплуатации наступает при принятии организацией решения о прекращении использования программного обеспечения. Такое решение может быть принято, например, при замене продукта другим решением, при прекращении оказания услуги или по иным внутренним основаниям.

Вывод из эксплуатации выполняется в следующем порядке:

11. Пользователи заблаговременно уведомляются о планируемом прекращении работы продукта.
12. Выполняется архивирование данных: оперативные данные из PostgreSQL, аналитические данные из ClickHouse и файлы из объектного хранилища S3 сохраняются в резервные копии для последующего хранения.
13. Прекращается доступ пользователей к программе.
14. Освобождаются вычислительные ресурсы: останавливается и удаляется развертывание в среде Kubernetes, при необходимости удаляются образы из реестра контейнеров.
15. Исходный код и история изменений сохраняются в системе контроля версий на случай возможного дальнейшего использования.

Порядок вывода из эксплуатации обеспечивает сохранность данных и исходного кода, корректное прекращение доступа пользователей и высвобождение задействованных ресурсов. Все действия на данной стадии выполняются ответственным сотрудником по внутреннему регламенту организации.

9. Роли и ответственность участников

В процессах жизненного цикла программного обеспечения «FinMe» участвуют сотрудники организации, за которыми закреплены определенные зоны ответственности. Перечень основных ролей и их зон ответственности приведен в таблице ниже.

Роль	**Зона ответственности**
Руководитель организации (правообладатель)	Принятие решений о развитии продукта, определение приоритетов, утверждение выпуска новых версий, распоряжение исключительным правом на программу.
Ответственный разработчик	Внесение изменений в исходный код, работа с ветками репозитория, объединение изменений, участие во внутреннем просмотре кода, выпуск и развертывание обновлений.
Специалист по тестированию и контролю качества	Проведение ручного тестирования по проверочным сценариям, выполнение проверочных (smoke) проверок, контроль результатов автоматических тестов и статического анализа.
Специалист поддержки пользователей	Прием и обработка обращений пользователей через мессенджер и электронную почту, регистрация задач в Яндекс Трекере, первичная реакция на обращения в течение рабочего дня.

Специалист по сопровождению инфраструктуры	Обслуживание сервера контроля версий GitLab, реестра контейнеров и среды развертывания Kubernetes, обеспечение сохранности данных и резервного копирования.

Приведенное распределение ролей является функциональным: в организации несколько ролей могут быть закреплены за одним сотрудником. Такое распределение ответственности обеспечивает выполнение всех процессов жизненного цикла программного обеспечения на каждой его стадии.