

Описание процессов жизненного цикла

ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

«АБиоГрам.Геном» («ABioGram.WGS»)

Аннотация

Настоящий документ содержит описание процессов жизненного цикла программного обеспечения «АБиоГрам.Геном» («ABioGram.WGS»), включая формирование требований, проектирование, разработку, тестирование, выпуск версий, установку, эксплуатацию, обновление, устранение неисправностей и техническую поддержку.

Оглавление

Аннотация 2
Сокращения и термины 4
Введение 5
1. Планирование процессов жизненного цикла разработки 5
1.1. Модель жизненного цикла и итерации 5
1.2. Управление версиями и изменениями 6
2. Формирование требований и анализ задач 7
3. Проектирование и реализация 7
4. Тестирование и отладка 8
5. Эксплуатация и сопровождение 9
5.1. Техническая поддержка 9
5.2. Персонал, обеспечивающий работу на местах 10
5.3. Персонал, обеспечивающий техническую поддержку и развитие системы 10

Сокращения и термины

Термин	Определение
Жизненный цикл ПО	Последовательность процессов от возникновения потребности и формирования требований до прекращения эксплуатации.
FASTA	Текстовый формат хранения нуклеотидных и иных биологических последовательностей.
MLST	Мультилокусное типирование по последовательностям аллелей заданных генетических локусов.
ST	Сиквенс-тип, устанавливаемый по сочетанию аллелей в конкретной схеме MLST.
Референсная база	Набор последовательностей и связанных описаний, по которым проводится аннотирование или типирование.
API / backend	Серверный программный интерфейс и обслуживающая его часть приложения.
Frontend	Пользовательская часть веб-приложения, открываемая в браузере.
Worker	Фоновый вычислительный обработчик задания аннотирования, типирования или подготовки базы.
Релиз	Идентифицируемая версия ПО, подготовленная к передаче и эксплуатации.
Smoke-тестирование	Краткая проверка основных функций после сборки или развёртывания.
Регрессионное тестирование	Проверка ранее работавших функций после внесения изменений.
Инцидент	Сбой или отклонение работы ПО, требующее регистрации и анализа.

Введение

«АБиоГрам.Геном» («ABioGram.WGS») представляет собой веб-приложение для хранения и систематизации FASTA-файлов микроорганизмов, ведения референсных баз последовательностей и выполнения молекулярно-биологического анализа. Для каждого образца могут храниться идентификаторы, вид микроорганизма, источник выделения, дата и место взятия материала, а также дополнительные пользовательские поля.

Программное обеспечение позволяет аннотировать последовательности по выбранным нуклеотидным и комбинированным базам, выполнять MLST-типирование, изучать характеристики выравниваний и координаты найденных участков, отображать график генов, формировать аналитические выборки и экспортировать результаты. Для авторизованных пользователей предусмотрены собственные коллекции. Для анонимной обработки используются доступные публичные базы.

Описание жизненного цикла охватывает взаимосвязанные программные компоненты и данные: веб-интерфейс, API, базу данных, фоновые обработчики, файловое хранилище, референсные данные и их версии. При сопровождении необходимо обеспечивать согласованность этих компонентов и воспроизводимость результатов вычислений.

1. Планирование процессов жизненного цикла разработки

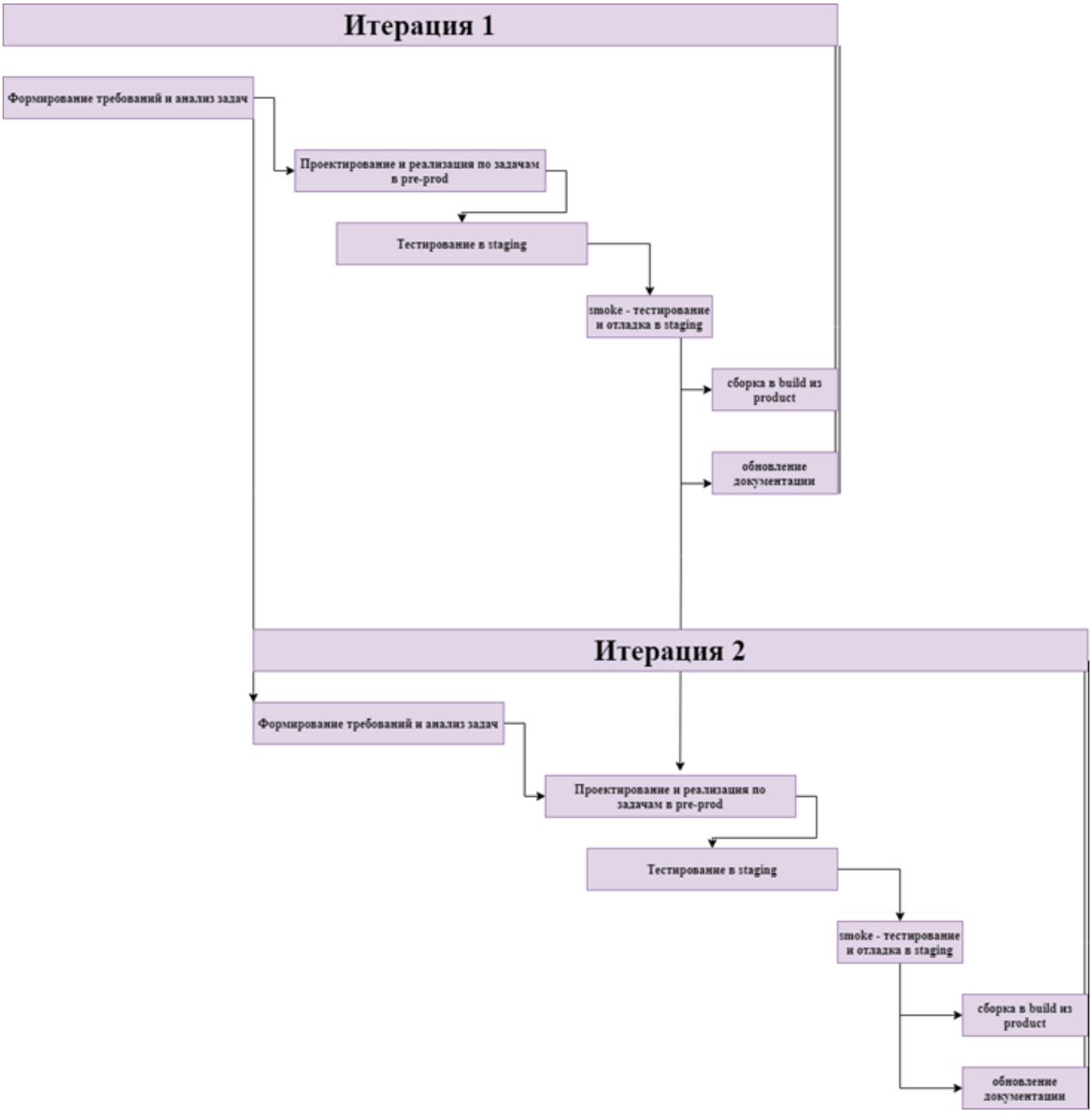
1.1. Модель жизненного цикла и итерации

Жизненный цикл (ЖЦ) включает период создания и использования «АБиоГрам.Геном» («ABioGram.WGS»), начиная с момента возникновения потребности в продукте, заканчивая разработкой, тестированием и отладкой, поставкой программной продукции, ее эксплуатацией на объектах Заказчика и технической поддержкой.

Жизненный цикл определен с учетом положений следующих стандартов:

- ГОСТ Р ИСО/МЭК 12207-2010 «Информационная технология. Системная и программная инженерия. Процессы жизненного цикла программных средств»;
- ГОСТ Р 56939-2016 «Защита информации. Разработка безопасного программного обеспечения. Общие требования»;
- ГОСТ Р ИСО 9001-2015 «Системы менеджмента качества. Требования».

Используется итерационная модель жизненного цикла (рисунок 1).



Выпуск новой версии продукта ведется в среднем каждые две недели согласно итерационной модели. Итерация включает этапы:

- формирования требований и оценку задач;
- проектирование и реализацию по задачам;
- тестирование и отладку;
- smoke-тестирование;
- обновление документации.

Процессы управления конфигурацией сервиса проводятся с использованием репозитория дистрибутивов, сервера сборки и системы контроля версий.

1.2. Управление версиями и изменениями

Для каждой планируемой версии фиксируется перечень требований и исправлений, ответственных исполнителей, затрагиваемые программные компоненты, изменения структуры данных, зависимости от референсных баз и результаты проверок. Изменение

принимается в релиз только после оценки последствий для действующих пользователей и данных. Управление версиями включает в себя:

- управление исходным кодом через систему контроля версий; идентификацию сборки по ревизии кода и версии контейнерных образов.
- проверку изменений в отдельном окружении до установки на рабочий экземпляр.
- фиксацию совместимых версий frontend, backend, worker и базы данных.
- сохранение записи о тестировании, причинах выпуска, обновлённой документации и способе возврата к работоспособной конфигурации.

2. Формирование требований и анализ задач

В рамках формирования требований и анализа задач выполняется следующий порядок задач:

- аналитик определяет задачи и требования к задачам, которые следует выполнить в итерацию;
- технический менеджер, аналитик совместно с разработчиком проводят анализ задач;
- технический менеджер определяет ответственность за разработку, назначает задачи и сроки выполнения разработчику;
- технический менеджер проводит мониторинг процесса.

Управление задачами ведётся с использованием релизов.
Не включенные в релиз задачи остаются в бэклоге и не участвуют в разработке, а используются для планирования будущих работ.
Результаты этапа:

1. Сформирован бэклог на итерацию и релиз;
2. Составлена спецификация требований и/или техническое описание по каждой задаче;
3. Назначены задачи и сроки выполнения.

3. Проектирование и реализация

В рамках проектирования и реализации выполняется следующий порядок задач:

- разработка программной архитектуры и решений по построению всех составных компонент;
- разработка исходных текстов, написание файлов спецификации для сборки программного обеспечения;
- сборка программного обеспечения и добавление в репозиторий программного обеспечения;
- сборка дистрибутивов из репозитория программного обеспечения.

Разработка по задачам идёт до тех пор, пока все задачи в рамках итерации не будут закрыты.
При работе с кодом команда придерживается методологии, известной как гитлаб флоу (https://about.gitlab.com/topics/version-control/what-is-gitlab-flow/).
В каждом репозитории есть три основных ветки и соответствующих им окружения:

- Мастер (с англ. Master, в терминологии гитлаб флоу) - ветка последнего стабильного релиза;
- Пре-продакшн (с англ. pre-production , в терминологии гитлаб флоу) - ветка для разработки, отправки на тестирование и готовящегося релиза;

- Продакшн (с англ. production, в терминологии гитлаб флоу) - ветка окончательного релиза.

4. Тестирование и отладка

Выполнение тестирования является обязательным перед передачей новой версии потребителю. Тестирование проводится лицом, ответственным за проведение тестирования программной продукции (инженер-тестировщик). Для тестирования и отладки продукции выделяется сервер, выдается задание на тестирование. По результатам тестирования осуществляется устранение ошибок и осуществляется (при необходимости) доработка программного обеспечения. Процессом тестирования и отладки определен следующий порядок:

- сборка дистрибутивов программного обеспечения - ответственный инженер по внедрению проводит сборку в pre-production окружении;
- проведение тестирования программного обеспечения - ответственный инженер-тестировщик проводит тестирование в staging, а также smoke-тестирование в staging всего продукта;
- устранение выявленных недостатков программного обеспечения - в случае обнаружения недостатков инженер-тестировщик формирует задачу с описанием дефекта (недостатка ПО), далее проводится повторное smoke-тестирование (ретест);
- при завершении smoke-тестирования проводится регрессионное тестирование всей функциональности продукта;
- при положительном результате регрессионного тестирования проводится добавление в репозиторий эталонных версий дистрибутивов и исходных текстов программного обеспечения;
- ответственный инженер проводит слияние pre-production в master и сборку (building) в production окружении;
- корректировка программной документации.

Тестирование выполняется до передачи новой версии в эксплуатацию. Оно включает проверку отдельных изменений, интеграции компонентов, выполнение основных пользовательских сценариев, регрессионное тестирование затронутых функций и контрольное развёртывание. Для тестового окружения используются данные с заранее известными ожидаемыми результатами.

Область проверки	**Минимальный контрольный сценарий**
Доступ и роли	Вход, выход, создание обычного пользователя, разграничение административных разделов; отдельно проверяется особенность имени root в интерфейсе.
FASTA и метаданные	Загрузка файла .fasta/.fas, создание карточки, заполнение полей, повторное чтение, удаление и экспорт.
Референсные базы	Загрузка NUC и NUC+AA, подготовка индексов, сопоставление заголовков, подтверждение готовности, контрольное аннотирование.

Аннотирование	Выполнение задания по заданной базе и порогам, сравнение генов, координат, % ID, выравнивания; различение отсутствия совпадений и ошибки.
MLST	Выбор подходящей схемы, проверка локусов, аллелей, ST/CC и случая неопределённого ST.
Результаты	Валидация и снятие отметки, график генов, выгрузки Excel, ZIP, PNG и SVG.
Дополнительные функции	Ввод MIC/DISK/SIR, аналитические выборки, фильтры, группировки и карта.
Анонимный режим	Загрузка по публичной базе, получение ссылки и результата, очистка по действующим настройкам, почтовое уведомление при настроенном SMTP.
Техническая устойчивость	Проверка очередей, работы backend/worker, действий при ошибке, резервного копирования и восстановления на отдельном стенде.

Результаты тестирования фиксируются с указанием версии программы, контрольной базы, параметров, набора данных и фактического исхода.

5. Эксплуатация и сопровождение

5.1. Техническая поддержка

Техническая поддержка пользователей осуществляется в формате консультирования пользователей и администраторов сервиса по вопросам установки, переустановки, инцидентов, администрирования и эксплуатации программного обеспечения по электронной почте support@abiogram.ru.

В рамках технической поддержки сервиса оказываются услуги:

- помощь в настройке и администрировании программного обеспечения;
- помощь в установке обновлений программного обеспечения;
- помощь в поиске и устранении проблем в случае некорректной установки обновления программного обеспечения;
- описание функционала программного обеспечения, помощь в эксплуатации;
- предоставление актуальной документации по настройке/работе программного обеспечения.

В заявке на техническую поддержку пользователь должен указать следующую информацию:

- описание проблемы;
- предпринятые попытки решения проблемы;
- релевантная дополнительная информация.

Завершенный запрос переходит в состояние закрытого после получения подтверждения от пользователя о решении запроса. В случае отсутствия ответа пользователя о завершении запроса в течение 14 рабочих дней, в случае если иное не

оговорено в соглашении о расширенной технической поддержки, запрос считается закрытым. Закрытие запроса может инициировать пользователь, если необходимость в ответе на запрос по каким-либо причинам более не требуется.

5.2. Персонал, обеспечивающий работу на местах

Пользователи ПО «АБиоГрам.Геном» («ABioGram.WGS») должны обладать навыками работы с персональным компьютером на уровне пользователя. Для работы с Системой пользователю необходимо изучить руководство пользователя «АБиоГрам.Геном» («ABioGram.WGS»). Администратор Системы должен владеть навыками работы с персональным компьютером на уровне уверенного пользователя. Обязательно знание основ работы вычислительной техники и программного обеспечения в локальных сетях, а также настроек системной политики прав пользователей в операционных системах семейства Windows и Linux.

5.3. Персонал, обеспечивающий техническую поддержку и развитие системы

Специалисты, обеспечивающие техническую поддержку и развитие Системы, должны обладать следующими знаниями и навыками:

1. Владение персональным компьютером на уровне уверенного пользователя;
2. Знание функциональных возможностей Системы и особенностей работы с ними;
3. Знание языков программирования;
4. Знание реляционных БД;
5. Значение протоколов обмена данных;
6. Знание средств восстановления баз данных.