ЗАЩИЩЕННАЯ СРЕДА РАЗРАБОТКИ И ИСПОЛНЕНИЯ «JAVA AT»

Описание поддержки жизненного цикла

Листов 40

2026

СОДЕРЖАНИЕ

1. ВВЕДЕНИЕ

Настоящий документ является описанием поддержки жизненного цикла и системы управления конфигурацией (далее – УК) программного обеспечения «Защищенная среда разработки и исполнения «Java AT»» (далее – средство, объект оценки (ОО)).

Настоящий документ подготовлен в рамках выполнения требований к средствам разработки, применяемым для создания средства, УК средства и разработке документации по безопасной разработке средства[1]. Сопоставление предъявляемых требований с описанием их реализации приведено в таблице 1.

В соответствии с документом «Программное обеспечение «Защищенная среда разработки и исполнения «Java AT». Технические условия», ОО является программным обеспечением, поэтому требования к аппаратной платформе средства не предъявляются.

Таблица 1 – Соответствие требований разделам документа

Содержание требования	**Раздел документа**
Требования к средствам разработки, применяемым для создания средства	
На выбранные средства разработки, применяемые для создания средства, должна быть разработана документация, включающая описания:	
Средств разработки, применяемых для создания средства	3
Использованных опций средств разработки, применяемых для создания средства	3
Требования к управлению конфигурацией средства	
К управлению конфигурацией средства предъявляются следующие требования:	
Управление изменениями средства и документации и обеспечение их уникальной маркировки	4, 5
Управление изменениями частей (элементов, компонентов) средства	4
Обеспечение уникальной идентификации всех элементов конфигурации	5
Управление изменениями средства, в том числе изменениями исходных текстов программного обеспечения средства	4
Применение автоматизированных мер контроля, обеспечивающих внесение в элементы конфигурации только санкционированных изменений	4
Организация процедур приемки модифицированных или вновь созданных элементов конфигурации	4

[1] Приказ ФСТЭК России от 02.06.2020 № 76 «Об утверждении Требований по безопасности информации, устанавливающих уровни доверия к средствам технической защиты информации и средствам обеспечения безопасности информационных технологий»

Содержание требования	Раздел документа
Управление изменениями сведений о недостатках безопасности и стадии их устранения	6
Регистрация всех изменений в средстве с использованием автоматизированных средств с указанием лица, инициирующего изменение, а также даты и времени регистрации изменений	4
Принятие мер, направленных на недопущение назначения ответственным за приемку элемента конфигурации лица, являющегося разработчиком средства	4
Применение автоматизированных средств идентификации всех элементов конфигурации, на которых оказывает влияние изменение какого-либо конкретного элемента конфигурации	4
Идентификация версии документации, представляемой для проведения оценки соответствия (сертификации), содержащей сведения о составе средства, в том числе сведения об исходных текстах программного обеспечения средства	5
Документация по управлению конфигурацией средства должна включать:	
Описание уникальной маркировки средства	5
Список элементов конфигурации средства, включающий, в том числе документацию	3
Порядок управления изменениями средства и документации	4
Описание метода, используемого для уникальной идентификации элементов конфигурации	5
Описание уникальных идентификаторов всех элементов конфигурации	5
Части (элементы, компоненты) средства в списке элементов конфигурации	3
Для каждого элемента конфигурации в списке элементов конфигурации должен быть указан разработчик	3
Сведения о составе средства, в том числе сведения об исходных текстах средства, в списке элементов конфигурации	3
Описание автоматизированных мер контроля, которые применяются для обеспечения внесения в элементы конфигурации только санкционированных изменений	4
План управления конфигурацией, содержащий описание процедур, используемых для приемки модифицированных или вновь созданных элементов конфигурации	4
Сведения о недостатках (уязвимостях) средства и стадии их устранения	6
Описание процедур регистрации всех изменений в средстве с использованием автоматизированных средств с указанием лица, инициирующего изменение, а также даты и времени регистрации изменений	4
Описание процедур, обеспечивающих контроль за осуществлением приемки элемента конфигурации лицом, не являющимся разработчиком средства	4
Описание процедур, обеспечивающих применение автоматизированных средств идентификации всех элементов конфигурации, на которых оказывает влияние изменение какого-либо конкретного элемента конфигурации	4
Описание процедур, обеспечивающих возможность идентификации версии документации, представляемой для проведения оценки соответствия (сертификации), содержащей сведения о составе средства, в том числе сведения об исходных текстах программного обеспечения средства	5
Сведения об инструментальных средствах разработки	3

Содержание требования	Раздел документа
Требования к разработке документации по безопасной разработке средства	
К разработке документации по безопасной разработке средства предъявляются следующие требования:	
Описание всех физических, процедурных, организационных и других мер безопасности, применяемых в среде разработки средства для защиты конфиденциальности и целостности проектной документации и реализации средства	7
Применяемые меры безопасности, направленные на снижение вероятности возникновения в средстве уязвимостей и иных недостатков, и их обоснование	7

2. ОБЩИЕ СВЕДЕНИЯ

Наименование средства: Программное обеспечение «Защищенная среда разработки и исполнения «Java AT».

Обозначение средства: RU.PCHT.00006-01 01.

Тип средства: защищенное средство обработки информации.

Краткое наименование средства: ПО «Java AT».

Разработчик и изготовитель: общество с ограниченной ответственностью «Эдвансед Трансформейшн Консалтинг» (ООО «ЭйТи Консалтинг») (127015, г. Москва, ул. Большая Новодмитровская, д. 14, стр. 7, офис 523, телефон: +7(495)748 05 75, сайт: https://www.at-consulting.ru/, электронная почта: clients@at-consulting.ru), имеющее лицензию на деятельность по разработке и производству средств защиты конфиденциальной информации рег. номер Л050-00107-00/00580843 (выдана ФСТЭК России 26.04.2012, действительна бессрочно) и лицензию на деятельность по технической защите конфиденциальной информации рег. номер Л024-00107-00/00581508 (выдана ФСТЭК России 15.03.2012, действительная бессрочно).

Заявитель на проведение сертификации: общество с ограниченной ответственностью «Эдвансед Трансформейшн Консалтинг» (ООО «ЭйТи Консалтинг») (127015, г. Москва, ул. Большая Новодмитровская, д. 14, стр. 7, офис 523, телефон: +7(495)748 05 75, сайт: https://www.at-consulting.ru/, электронная почта: clients@at-consulting.ru), имеющее лицензию на деятельность по разработке и производству средств защиты конфиденциальной информации рег. номер Л050-00107-00/00580843 (выдана ФСТЭК России 26.04.2012, действительна бессрочно) и лицензию на деятельность по технической защите конфиденциальной информации рег. номер Л024-00107-00/00581508 (выдана ФСТЭК России 15.03.2012, действительная бессрочно).

3. СПИСОК КОНФИГУРАЦИИ СРЕДСТВА

Система УК отслеживает следующие элементы конфигурации, входящие в состав средства:

- ОО;
- составные части ОО;
- представление реализации ОО;
- документацию на ОО;
- инструментальные средства разработки и связанную с ними информацию.

3.1. Средство и его составные части

Перечень компонентов, входящих в состав средства представлен в документе «Программное обеспечение «Защищенная среда разработки и исполнения «Java AT»». Описание программы» (раздел 3).

Разработчик компонентов – ООО «ЭйТи Консалтинг».

3.2. Представление реализации средства

3.2.1. Модули средства

В состав реализации средства входят модули, приведенные в таблице 2, где представлено:

- описание модулей ОО;
- для модулей ОО, реализующих функции безопасности, представлено описание интерфейсов, возвращаемых ими в ответ на запросы значений, взаимодействий с другими модулями и вызываемыми интерфейсами этих модулей;
- для модулей ОО, не влияющих на выполнение функций безопасности – описание назначения и взаимодействия с другими модулями.

Таблица 2 – Описание модулей ОО

Название модуля	Описание модуля
java	Модуль предназначен для запуска виртуальной машины и передачи ей аргументов, поданных через командную строку.
libjvm.so	Модуль предназначен для реализации всей логики виртуальной машины Java. Модуль выполняет следующие функции: - загрузку классов; - исполнение прикладных программ; - компиляция Java-байткода; - управление памятью; Модуль реализует меры защиты информации: УПД.2, РСБ.3, ОЦЛ.1. Модуль выполняет следующие функции безопасности: - Поддержка управления доступом субъектов доступа к объектам доступа - Регистрация событий безопасности - Очистка памяти - Изоляция программных модулей - Безопасная интерпретация байткода - Независимость экземпляров виртуальных машин - Неизменность исполняемого кода
modules	Модуль предназначен для реализации логики библиотеки классов JDK в части реализации на языке Java. Модуль выполняет следующие функции: - загрузка классов; - исполнение прикладных программ. Модуль реализует меры защиты информации: УПД.2, РСБ.3. Модуль выполняет следующие функции безопасности: - Поддержка управления доступом субъектов доступа к объектам доступа - Регистрация событий безопасности - Изоляция программных модулей - Неизменность исполняемого кода

3.2.2. Информация о соответствии модулей и подсистем средства

Информация о соответствии модулей безопасности и подсистем ОО представлена в таблице 3.

Таблица 3 – Информация о соответствии модулей и подсистем ОО

Название подсистемы ОО	Перечень модулей
Виртуальная машина Java	java libjvm.so
Библиотека классов JDK	modules

3.2.3. Информация о прослеживании интерфейсов к модулям средства

Информация о прослеживании интерфейсов к модулям ОО представлена в таблице 4.

Таблица 4 – Информация о прослеживании интерфейсов к функциям безопасности и модулей ОО

Наименование модуля	Связный ИФБО
java	ИФБО.1.1 Опция «-Dprotected.java.root»; ИФБО.1.2 Опция «-jar»; ИФБО.1.3 Опция «-cp / -classpath / --class-path»; ИФБО.1.4 Опция «-m / --module»; ИФБО.1.5 Опция «-p / --module-path»; ИФБО.1.6 Опция «--upgrade-module-path»; ИФБО.1.7 Опция «--add-modules»; ИФБО.1.8 Опция «--enable-native-access». ИФБО.1.9 Опция «--enable-preview»; ИФБО.1.10 Опция «-agentlib»; ИФБО.1.11 Опция «-agentpath»; ИФБО.1.12 Опция «-javaagent»; ИФБО.1.16 Опция «-D»; ИФБО.1.17 Опция «-Xlog»; ИФБО.1.18 Опция «-XX:»; ИФБО.1.19 Опция «--parsec-privsock»; ИФБО.1.20 Опция «--parsec-changepriv»; ИФБО.1.21 Опция «--parsec-setmac»; ИФБО.3.1 Стандартный вывод. ИФБО.4.1 Стандартный поток ошибок. ИФБО.5.1 Стандартный ввод
libjvm.so	ИФБО.1.1 Опция «-Dprotected.java.root»; ИФБО.1.2 Опция «-jar»; ИФБО.1.3 Опция «-cp / -classpath / --class-path»; ИФБО.1.4 Опция «-m / --module»; ИФБО.1.5 Опция «-p / --module-path»; ИФБО.1.6 Опция «--upgrade-module-path»; ИФБО.1.7 Опция «--add-modules»; ИФБО.1.8 Опция «--enable-native-access». ИФБО.1.9 Опция «--enable-preview»; ИФБО.1.10 Опция «-agentlib»; ИФБО.1.11 Опция «-agentpath»; ИФБО.1.12 Опция «-javaagent»; ИФБО.1.16 Опция «-D»; ИФБО.1.17 Опция «-Xlog»; ИФБО.1.18 Опция «-XX:»; ИФБО.1.19 Опция «--parsec-privsock»; ИФБО.1.20 Опция «--parsec-changepriv»; ИФБО.1.21 Опция «--parsec-setmac»;

Наименование модуля	Связный ИФБО
	ИФБО.2.1 Переменная окружения «_JAVA_OPTIONS»; ИФБО.2.2 Переменная окружения «JAVA_TOOL_OPTIONS»; ИФБО.2.3 Переменная окружения «LD_LIBRARY_PATH»; ИФБО.3.1 Стандартный вывод. ИФБО.4.1 Стандартный поток ошибок. ИФБО.5.1 Стандартный ввод
modules	ИФБО.1.1 Опция «-Dprotected.java.root»; ИФБО.1.2 Опция «-jar»; ИФБО.1.3 Опция «-cp / -classpath / --class-path»; ИФБО.1.4 Опция «-m / --module»; ИФБО.1.5 Опция «-p / --module-path»; ИФБО.1.6 Опция «--upgrade-module-path»; ИФБО.1.7 Опция «--add-modules»; ИФБО.1.8 Опция «--enable-native-access». ИФБО.1.9 Опция «--enable-preview»; ИФБО.1.16 Опция «-D»; ИФБО.1.19 Опция «--parsec-privsock»; ИФБО.1.20 Опция «--parsec-changepriv»; ИФБО.1.21 Опция «--parsec-setmac»; ИФБО.3.1 Стандартный вывод. ИФБО.4.1 Стандартный поток ошибок. ИФБО.5.1 Стандартный ввод

3.3. Инструментальные средства разработки

Для хранения исходных текстов продукта используется некоммерческий проект с открытым исходным кодом GitLab v18.4.1. Доступ к хранилищу осуществляется штатными инструментами среды разработки Git 2.20.1.

Средство разработано с использованием следующих языков программирования:

- Java (версии 20);
- C (версии C11)
- C++ (версии C++14).

3.3.1. Система учета запросов и ошибок

Для учета запросов и ошибок средства используется система учета запросов и ошибок, встроенная в систему GitLab, используемую для разработки ОО (раздел 3.3).

3.3.2. Средства сборки

Управление сборкой ОО осуществляется с помощью скриптов сборки, входящих в состав ОО. Сборка осуществляется с помощью bash-скрипта build.sh, расположенного в корневой директории исходных кодов.

Сборка производится на ОС Astra Linux с использованием следующих средств:

– Make 4.2.1;

– Autoconf 2.69.

3.3.3. Средства разработки

Среда разработки: Visual Studio Code.

Компилятор C++: gcc (version 8.3.0).

Компилятор Java: javac (version 21.0.6).

3.3.4. Средства тестирования

Для тестирования ОО используются следующие средства:

– GCOV (8.3.0);

– JCOV (3.0);

– LibFuzzer (11.0.1);

– Jazzer (0.23.0);

– SpecJVM2008 (1.0.1);

– SpecJBB2005 (1.07);

– SpecJVM98 (1.03);

– Dacapo (9-12-MR1);

– JCTF (v8);

– JTREG (8.1).

3.3.5. Средства хранения проектной документации

Для хранения проектной документации используется некоммерческий проект с открытым исходным кодом OnlyOffice (v12.5.2.1848). Доступ к хранилищу осуществляется штатными инструментами через веб-интерфейс.

3.3.6. Опции компиляции и сборки

Для каждого средства разработки применяются следующие опции и настройки, обеспечивающие безопасность:

- Компиляция C++ (gcc):
  - D__STDC_FORMAT_MACROS
  - D__STDC_LIMIT_MACROS
  - D__STDC_CONSTANT_MACROS
  - D_GNU_SOURCE
  - D_REENTRANT
  - pipe
  - fno-rtti
  - fno-exceptions
  - fvisibility=hidden
  - fno-strict-aliasing
  - fno-omit-frame-pointer
  - fstack-protector
  - std=c++14
  - DLIBC=gnu
  - DLINUX
  - DDO_MANGLING
  - DINTEGRITY
  - Wall
  - Wextra

- Wformat=2
- Wpointer-arith
- Wsign-compare
- Wunused-function
- Wundef
- Wunused-value
- Wreturn-type
- Wtrampolines
- Woverloaded-virtual
- Wreorder
- fPIC
- fno-delete-null-pointer-checks
- fno-lifetime-dse
- Wno-format-zero-length
- Wtype-limits
- Wuninitialized
- m64
- Wno-unused-parameter
- Wno-unused
- Wno-array-bounds
- Wno-comment
- Wno-delete-non-virtual-dtor
- Wno-empty-body
- Wno-ignored-qualifiers
- Wno-implicit-fallthrough
- Wno-int-in-bool-context
- Wno-maybe-uninitialized
- Wno-missing-field-initializers

  - Wno-parentheses
  - Wno-shift-negative-value
  - Wno-unknown-pragmas
  - O3
- Компиляция C (gcc):
  - pipe
  - fstack-protector
  - DLIBC=gnu
  - D_GNU_SOURCE
  - D_REENTRANT
  - D_LARGEFILE64_SOURCE
  - DPARSEC_SUPPORT
  - DPROTECTED_JAVA_ROOT
  - DINTEGRITY
  - DLINUX
  - DNDEBUG
  - std=c11
  - fno-strict-aliasing
  - Wall
  - Wextra
  - Wformat=2
  - Wpointer-arith
  - Wsign-compare
  - Wunused-function
  - Wundef
  - Wunused-value
  - Wreturn-type
  - Wtrampolines

    - m64
    - fno-omit-frame-pointer
    - fno-delete-null-pointer-checks
    - fno-lifetime-dse
    - fPIC
    - fvisibility=hidden
    - Wno-unused-parameter
    - Wno-unused
    - O3
- Линковка C++ (gcc):
    - Wl,-z,defs
    - Wl,-z,relro
    - Wl,-z,now
    - Wl,-z,noexecstack
    - Wl,--hash-style=gnu
    - m64
    - static-libstdc++
    - static-libgcc
    - shared
    - O1
- Линковка
    - Wl,-z,defs
    - Wl,-z,relro
    - Wl,-z,now
    - Wl,-z,noexecstack
    - Wl,-O1
    - m64

- Компиляция Java (javac):
  - g
  - Xlint:all
  - source 20
  - target 20
  - Xlint:-options
  - implicit:none
  - Xprefer:source
- Git:
  - настроена двухфакторная аутентификация для доступа к репозиторию;
  - интеграция с CI/CD для автоматического анализа при каждом коммите.

4. ПЛАН УПРАВЛЕНИЯ КОНФИГУРАЦИЕЙ СРЕДСТВА

В системе управления конфигурацией средства предусмотрено выполнение следующих операций над элементами конфигурации:

- создание элемента конфигурации;
- модификация элемента конфигурации.

Операция удаления элемента конфигурации не используется.

4.1. Программное обеспечение управления конфигурацией

УК осуществляется с использованием следующих систем:

- система хранения исходных текстов, управления версиями и тестирования продуктов Git. Фиксирует изменения в исходном коде с указанием даты, времени и имени пользователя, внесшего правки. Позволяет руководителю проекта просматривать изменения, затем подтверждать или отменять внесенные правки. Также система позволяет обеспечить управление доступом к репозиторию с исходными текстами проекта на основе учетных записей;
- сервер для хранения документации с установленным сервисом OnlyOffice позволяет задавать права доступа ответственным пользователям только к необходимым ресурсам;
- система планирования работ, учета запросов и ошибок GitLab. Позволяет создавать и назначать задачи по выполнению операций над элементами конфигурации, отрабатывать возникающие ошибки, контролировать ход выполнения работ, а также хранит информацию о событиях по проекту.

4.2. Уникальная идентификация элементов конфигурации

Git позволяет контролировать состав файлов исходных текстов и других исходных модулей средства и обеспечивать их уникальную поверсионную идентификацию, что гарантирует внесение в элементы конфигурации только санкционированных изменений.

Система уникальной идентификации элементов конфигурации в Git строится на основе меток. Метка представляет собой буквенное имя версии, которое присваивается как дополнительный идентификатор зафиксированной версии образа файловой системы репозитория.

Метки создаются в ключевых точках развития проекта, таких как выпуск новой версии. Использование меток позволяет обеспечить однозначное соответствие версии программного продукта и версий файлов, входящих в ее состав. Git не содержит жестких правил, регламентирующих, как должны быть устроены имена меток. Соглашение о способах наименования меток относится к компетенции разработчика.

4.3. Контроль доступа к управлению конфигурацией

Для контроля доступа к системе хранения исходных кодов используются средства идентификации, аутентификации и разграничения доступа, встроенные в систему контроля версий Git.

Для каждого разработчика установлены учетные записи в системе контроля версий Git. Для каждой учетной записи установлены права доступа на определенные проекты с исходными текстами. Аутентификация разработчиков производится с использованием пароля.

Система позволяет фиксировать изменения в исходном коде с указанием даты, времени и имени пользователя, внесшего правки, подтверждать или отменять внесенные правки, выбирать необходимую версию файлов, с учетом изменений, локально изменять и применять изменения к требуемым файлам.

Для контроля доступа к системе хранения электронных документов используются средства идентификации, аутентификации и разграничения доступа для системы хранения документации OnlyOffice. Администратор системы задает права доступа ответственных пользователей только к необходимым ресурсам в соответствии с должностными обязанностями.

Для контроля доступа к системе планирования используется система контроля доступа из состава GitLab.

Разграничение доступа сотрудников к разделам осуществляется в соответствии с их полномочиями. Права на администрирование имеются только у администраторов системы.

Кроме того, составной частью системы управления проектами GitLab является система учета запросов и ошибок. Ошибки и запросы помещаются в список с указанием важности, сроков исполнения и ответственного лица. Возможны также комментарии для уточнения информации по конкретному элементу списка.

4.4. План приемки модифицированных или созданных элементов конфигурации средства

4.4.1. Общий порядок

Инициатором процесса модификации существующих или создания новых элементов конфигурации средства является заявитель. Причиной для модификации существующих или создания новых элементов конфигурации средства являются следующие условия:

- обнаружен недостаток в элементах конфигурации средства;
- произошли изменения в требованиях по безопасности информации, предъявляемых к средству;
- возникла необходимость внесения изменений в функциональные возможности средства.

4.4.2. Модификация и приемка представления реализации средства

При возникновении необходимости внесения изменений в исходный код средства только руководитель проекта инициирует соответствующую задачу в системе Gitab с подробным описанием необходимого результата модификации и сроков исполнения задачи. При необходимости прикладываются дополнительные материалы. Созданная задача назначается для выполнения ответственному лицу из команды разработки. Разработчик приступает к реализации задачи,

при необходимости запрашивая дополнительные данные, описывая трудозатраты на выполнение задачи и степень ее готовности.

В соответствии со стадиями готовности периодически выполняется рефакторинг кода, хранимого в системе Git.

После завершения задачи разработчик изменяет ее статус, тем самым, переводя задачу на тестировщика, ответственного за составление набора тестов, проведение тестирования и оформление полученных результатов в виде отчета. По результатам тестирования при несоответствии реализованных функций заявленным руководителем проекта задача снова переводится на разработчика для корректировки исходного кода. Итерация повторяется, пока результаты проведенных этапов тестирования не позволят сделать вывод о соответствии новой сборки средства необходимым требованиям.

Версии сборки средства, соответствующей заявленным требованиям, присваивается идентификатор.

Любое изменение исходного кода влечет за собой обновление контрольной суммы файлов компонентов средства. Таким образом, модификация кода приводит к изменению элементов, относящихся к представлению реализации средства.

Если модификация исходного кода связана с преобразованием функциональных возможностей средства, то это приводит к изменению эксплуатационной документации, а также к добавлению элементов тестовой документации.

Если внесенные в исходный код изменения требуют корректировки документации, то данный процесс выполняется в соответствии с пунктом 4.4.3.

4.4.3. Модификация, создание и приемка документации

При возникновении необходимости разработки новых документов или внесения изменений в существующие, только руководитель проекта инициирует соответствующую задачу в системе GitLab с подробным описанием необходимого результата модификации или создания документов, а также сроков исполнения задачи. Созданная задача назначается для выполнения ответственному исполнителю

из числа разработчиков проекта. Исполнитель приступает к реализации задачи, при необходимости запрашивая дополнительные данные, описывая трудозатраты на выполнение задачи и степень ее готовности.

Модифицированный или вновь созданный документ направляется на верификацию руководителю проекта. При успешной верификации документа задача переназначается на технического писателя, выполняющего проверку оформления документа в соответствии с принятыми стандартами. Технический писатель при необходимости, вносит изменения в документ самостоятельно, либо создает подзадачи по корректировке и назначает их ведущему исполнителю.

Электронный экземпляр модифицированного или созданного документа размещается руководителем проекта в системе OnlyOffice проекта либо взамен неактуального, либо с сохранением его с пометкой об утрате актуальности.

Внесение изменений в документацию может привести к необходимости модификации исходного кода средства.

4.4.4. Регистрация событий создания и модификации элементов конфигурации

Процесс создания или модификации каждого элемента конфигурации средства начинается с постановки соответствующей задачи в системе GitLab. Задаче автоматически присваивается номер, фиксируется инициатор, дата и время создания. В процессе выполнения задачи фиксируется каждая итерация: создание подзадач, добавление комментариев, вложений, перевод задачи в различные статусы, изменение исполнителей и т. д.

Регистрация событий, связанных с модификацией элементов конфигурации, относящихся к представлению реализации, осуществляется системой Git, путем фиксации всех изменений в исходном коде с указанием даты, времени и имени пользователя, внесшего правки.

Регистрация событий, связанных с модификацией и созданием элементов конфигурации, относящихся к документации, осуществляется средствами журналирования системы управления документацией OnlyOffice.

5. ОПИСАНИЕ МЕТОДА, ИСПОЛЬЗУЕМОГО ДЛЯ УНИКАЛЬНОЙ ИДЕНТИФИКАЦИИ ЭЛЕМЕНТОВ КОНФИГУРАЦИИ

5.1. Роли и обязанности лиц, требуемые для выполнения операций на отдельных элементах конфигурации

В системе УК поддерживаются следующие роли сотрудников, требуемые для выполнения операций на отдельных элементах конфигурации:

- руководитель проекта;
- разработчик;
- тестировщик;
- системный администратор;
- технический писатель.

Руководитель проекта осуществляет общее управление конфигурацией проекта. В его обязанности входит:

- утверждение состава элементов конфигурации (основных средств и документации);
- контроль соблюдения регламентов управления конфигурацией и организационно-технических мер безопасности;
- управление доступом к элементам конфигурации для членов проектной команды;
- верификация и утверждение изменений в исходном коде и документации перед их включением в официальные версии проекта;
- санкционирование создания основных версий и релизов проекта.

Технический писатель управляет конфигурацией документации проекта. Его задачи включают в себя:

- разработка и внесение изменений в тексты документации в системе управления версиями;
- согласование и регистрация изменений в документации, связанных с модификацией других элементов конфигурации (например, при изменении функциональности средства);

– обеспечение соответствия версий документации версиям основного средства.

Разработчик управляет конфигурацией представления реализации средства (исходного кода). В его обязанности входит:

– внесение изменений в исходный код в выделенных рабочих потоках (ветках);
– регистрация изменений в системе управления версиями;
– формирование и оформление запросов на внесение изменений в основную версию кода;
– участие в анализе и решении о возможности интеграции изменений в другие элементы конфигурации;
– информирование руководителя проекта и технического писателя о внесенных изменениях в код для обеспечения актуальности связанных элементов конфигурации (например, документации).

Тестировщик управляет конфигурацией элементов, связанных с тестированием. В его обязанности входит:

– разработка и версионирование планов тестирования, тестовых сценариев и данных;
– фиксация результатов тестирования в системе управления конфигурацией;
– регистрация выявленных инцидентов и дефектов, их связь с версиями тестируемых компонентов;
– контроль соответствия версий тестовых артефактов версиям тестируемого средства.

Системный администратор обеспечивает инфраструктуру для управления конфигурацией, отвечая за:

– развертывание и поддержку работы систем управления версиями и хранения артефактов;
– резервное копирование и восстановление репозиториев конфигурации;
– управление доступом к инфраструктуре УК на техническом уровне.

5.2. Система маркировки

Устанавливается следующая структура обозначения ОО и документации:

A.B.XXXXX – TT NN QQ-D, где:

- A – код страны;
- B – код организации разработчика (или ОКПО);
- XXXXX – код вида продукции по классификатору ОКП ОК 005-93;
- TT – номер издания (для средства), номер редакции (для документа);
- NN – код вида документа;
- QQ – номер документа данного вида;
- D – номер части документа.

Пример: RU.32146099.501212-01 51 01, где

- RU – код страны;
- 32146099 – ОКПО разработчика;
- 501212 –код вида продукции по классификатору ОКП ОК 005-93 (Системное ПО - языки объектно-ориентированные);
- номер редакции;
- 51 – номер документа данного вида (Программа и методика испытаний);
- 01 – номер части документа.

Каждый элемент представления реализации средства маркируется следующим образом: XXX.R, где:

- XXX – наименование файла;
- R – расширение файла.

5.2.1. Верификация элементов конфигурации

Администратор имеет возможность верифицировать версию элементов конфигурации средства. В исходном коде версия среды разработки и исполнения «Java AT» указана в файле build.sh.

6. СВЕДЕНИЯ О НЕДОСТАТКАХ. УПРАВЛЕНИЕ ИЗМЕНЕНИЯМИ СВЕДЕНИЙ О НЕДОСТАТКАХ БЕЗОПАСНОСТИ, ВКЛЮЧАЯ УЯЗВИМОСТИ, И СТАДИИ ИХ УСТРАНЕНИЯ

Разработчик и изготовитель принимает на себя обязательства по устранению недостатков в ОО на протяжении всего жизненного цикла ОО.

Разработчик и изготовитель осуществляет прием сообщений о недостатках:

- по адресу: г. Москва, ул. Большая Новодмитровская, д. 14, стр. 7, офис 523;
- по телефону: +7 (495) 748 0575;
- по электронной почте: clients@at-consulting.ru.

Разработчик и изготовитель периодически, не реже одного раза в месяц, должен проводить поиск известных (подтвержденных) уязвимостей в общедоступных источниках информации об уязвимостях. В качестве общедоступных источников в первую очередь должны использоваться база данных уязвимостей в составе банка данных угроз безопасности информации ФСТЭК России (www.bdu.fstec.ru) (далее – БДУ), а также следующие дополнительные источники:

- https://cve.mitre.org/;
- https://nvd.nist.gov/;
- https://www.exploit-db.com/;
- http://www.rapid7.com/db/;
- http://www.cvedetails.com/;
- http://www.securitylab.ru/ и другие.

Разработчик и изготовитель должен провести анализ выявленных уязвимостей на предмет возможности их использования для нарушения безопасности. При анализе уязвимостей необходимо учитывать следующие критерии:

- тип уязвимости;
- версию программного обеспечения, подверженную уязвимости;
- уровни опасности уязвимости (критическая, высокая, средняя, низкая);
- информацию об устранении.

Процедура устранения уязвимостей ОО должна обеспечивать возможность обновления ОО для устранения актуальных уязвимостей.

В случае выявления информации об уязвимости ОО и сред его функционирования из различных источников и отсутствия информации об этой уязвимости в БДУ, разработчик и изготовитель предоставляет информацию о данной уязвимости во ФСТЭК России для размещения в БДУ.

Устранение недостатков должно предусматривать доведение информации о недостатках ОО, а также о компенсирующих мерах по защите информации или ограничениях по применению, а также доработку ОО, в том числе разработку обновлений ОО или разработку мер по защите информации, нейтрализующих недостаток. Общий срок устранения недостатка ОО не должен превышать 60 дней с момента выявления недостатка.

При выявлении уязвимостей ОО разработчик и изготовитель должен осуществить следующие мероприятия:

1. В случае отсутствия на момент проверки информации по выявленным уязвимостям ОО доступных релизов ОО с устраненными уязвимостями разработать компенсирующие меры по защите информации или ограничения по применению ОО, снижающие возможность эксплуатации уязвимостей, а также инструкцию по проведению организационно-технических мероприятий.

2. Довести информацию о компенсирующих мерах и ограничениях по применению до потребителей в срок не более 48 часов с момента выявления недостатка.

3. Доработать ОО или его отдельные компоненты, в том числе выпустить обновление ОО или, в случае невозможности устранения уязвимостей ОО путем применения обновления, выпустить меры по защите информации, нейтрализующие недостаток и внести необходимые изменения в эксплуатационную документацию.

4. Провести тестирование доработанного ОО или его отдельных компонентов на предмет влияния обновлений ОО на его функции безопасности, подтверждения устранения уязвимостей, невнесения новых уязвимостей в ОО.

5. Довести информацию о недостатках ОО, о компенсирующих мерах по защите информации или ограничениях по применению до каждого потребителя путем отправки сообщений на электронные адреса потребителей или за счет

применения компонента средства, обеспечивающего доведение указанной информации до потребителя автоматически.

6. Довести информацию о недостатках ОО, а также о компенсирующих мерах по защите информации или ограничениях по применению до каждого потребителя способом, обеспечивающим подлинность и целостность доводимой информации.

7. Обеспечить гарантированную доставку обновлений ОО потребителям.

8. Если уязвимость не устраняется обновлением ОО или реализацией мер по защите информации, нейтрализующих недостаток, незамедлительно и гарантированно, с подтверждением, сообщить об этом всем потребителям и во ФСТЭК России. После данного сообщения потребители должны прекратить применение ОО.

Если потребитель не может выполнить установку обновления ОО и/или реализовать меры по защите информации, нейтрализующие недостаток ОО, он прекращает его применение.

7. МЕРЫ БЕЗОПАСНОСТИ

7.1. Физические меры

Помещения, в которых осуществляется процесс разработки средства, удовлетворяют санитарным нормам и правилам, требованиям безопасности труда и охраны окружающей среды. Обеспечивается круглосуточная охрана, контроль внешнего периметра и внутренних помещений (видеонаблюдение), действует пропускной режим.

Кабинеты оборудованы системой пожарной сигнализации. Ключи от входных дверей в кабинеты сдаются службе охраны по зданию, что регистрируется в журнале выдачи ключей с указанием времени сдачи и выдачи ключей.

Доступ посторонних лиц в помещения, где размещены технические средства, осуществляющие обработку конфиденциальной информации, а также хранятся носители информации ограничен.

Определен регламент по допуску в помещения, который включает в себя:

- механизмы доступа (исключения доступа) в защищаемые помещения;
- определение лиц, ответственных за контроль и управление доступом к техническим средствам, средствам защиты информации, средствам обеспечения функционирования, а также в помещения, в которых они установлены;
- порядок передачи информации за пределы контролируемой зоны.

7.2. Процедурные меры

Процедурные меры обеспечения безопасности, выполняемые в процессе разработки средства:

- разграничение прав доступа пользователей к техническим средствам;
- резервное копирование информации;
- незамедлительная блокировка учетной записи сотрудника при компрометации (или подозрении на компрометацию).

7.3. Организационные меры

Работники принимают участие в реализации всех мер по разработке безопасного ПО.

Обязанности работников в области создания безопасного ПО распространяются на следующие процессы:

- требования по безопасности к разрабатываемому (модернизируемому) ПО предъявляются руководителем соответствующего проекта, а также аналитиками;
- проектирование архитектуры программы выполняется аналитиком, совместно с руководителем проекта и разработчиками;
- конструирование и комплексирование ПО выполняется разработчиками;
- квалификационное тестирование выполняется разработчиками и тестировщиками ПО;
- инсталляцию и поддержку приемки ПО, а также маркировку версий выполняют работники, ответственные за подготовку комплекта ОО к поставке;
- в процессе эксплуатации ПО процедуры безопасной разработки выполняются работниками технической поддержки, разработчиками и тестировщиками;
- управление конфигурацией выполняют разработчики ПО;
- за подготовку программы обучения и инструктаж работников отвечают инструкторы; за анализ программы и результатов – директор практики «Информационная безопасность».

7.4. Технические меры безопасности

Сборка ОО производится на отдельной, выделенной для этой цели, аппаратной платформе. Доступ к аппаратной платформе имеют только конкретные разработчики на основе логического имени и пароля.

Рабочее место каждого сотрудника оборудовано персональным компьютером, подключенным к локальной сети. Для проведения тестовых испытаний разрабатываемых программных продуктов имеется тестовый стенд. Доступ к персональным компьютерам и средствам вычислительной техники, входящим в состав тестового стенда, осуществляется на основе логического имени и пароля пользователя в рамках операционных систем.

На рабочих местах каждого сотрудника применяются средства антивирусной защиты.

На границе сети применяются средства межсетевого экранирования.

7.5. Меры безопасности, направленные на снижение вероятности возникновения в средстве уязвимостей

В целях снижения вероятности возникновения уязвимостей обеспечивается:

- поиск ошибок реализации в средстве на протяжении всего жизненного цикла;
- поиск известных (подтвержденных) уязвимостей в общедоступных источниках информации об уязвимостях;
- информирование сотрудников о появлении актуальных угроз безопасности информации, обучение правилам безопасной разработки;
- контроль уровня знаний сотрудников по вопросам обеспечения защиты информации.

7.6. Выводы

Физические, процедурные, организационные и технические меры позволяют обеспечить безопасность разработки средства.

8. СВИДЕТЕЛЬСТВО ВЫПОЛНЕНИЯ ПЛАНА УПРАВЛЕНИЯ КОНФИГУРАЦИЕЙ СРЕДСТВА

8.1. Свидетельство выполнения процедур доступа и внесения изменений в элементы конфигурации средства

Для получения доступа к системе контроля версий программного проекта Git необходимо пройти аутентификацию и авторизацию в системе GitLab. Все попытки идентификации и аутентификации фиксируются в журналах аудита. В случае неуспешной идентификации и аутентификации на экране отображается сообщение «Could not authenticate you from Ldapmain because «Invalid credentials for test». Примеры неуспешного прохождения процедуры идентификации и аутентификации представлены на рисунке 1.

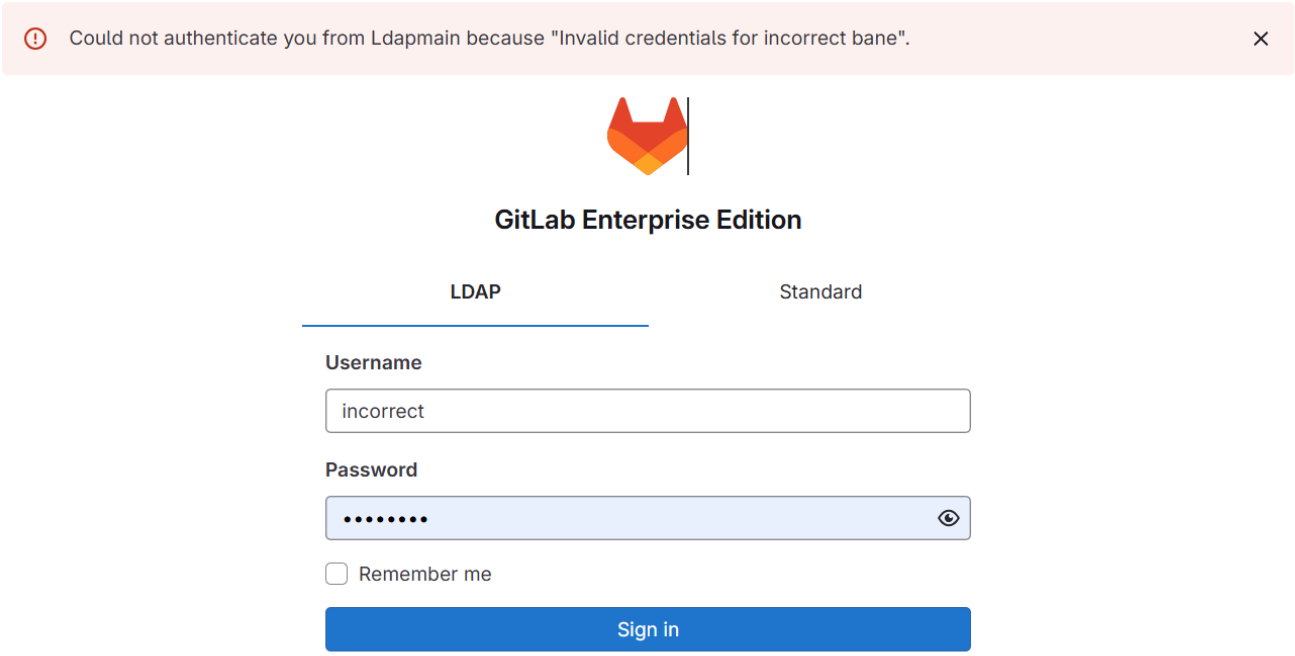

Рисунок 1 – Некорректный ввод уникального идентификатора или пароля пользователя

В случае успешной идентификации и аутентификации в системе GitLab пользователь взаимодействует с помощью ssh-ключа с репозиторием по протоколу Git. Пример главной страницы репозитория проекта в GitLab после успешной идентификации и аутентификации представлен на рисунке 2.

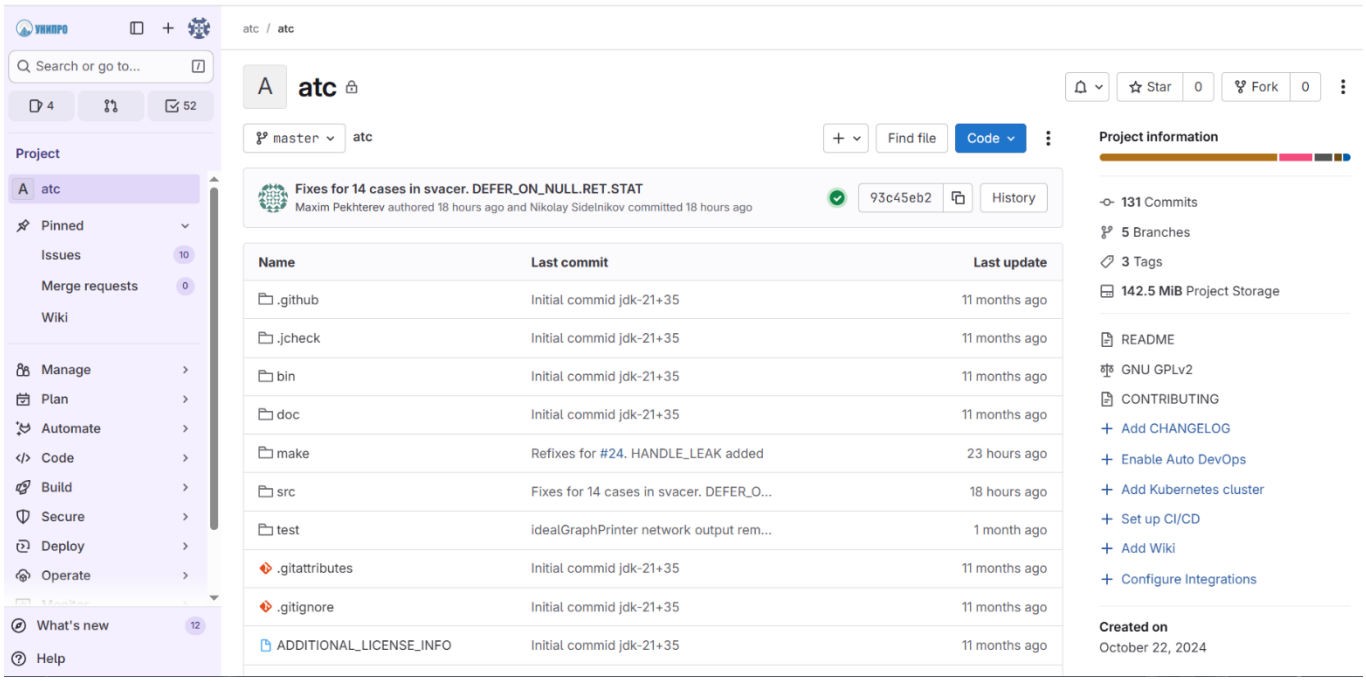

Рисунок 2 – Скриншот главной страницы репозитория в Git

Для каждого разработчика установлены учетные записи в системе контроля версий Git. Для каждой учетной записи установлены права доступа (Рисунок 3).

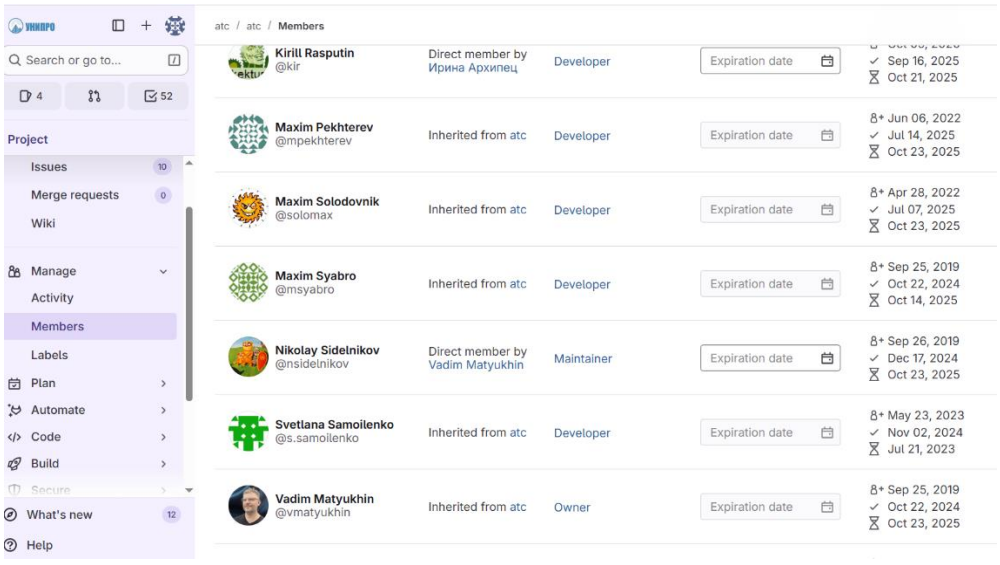

Рисунок 3 – Права доступа в системе Git

Доступ к системе учета запросов и ошибок осуществляется в соответствии с назначенными ролями в GitLab и представлен на рисунке 3.

Пример настройки ролей доступа к сервису управления документацией OnlyOffice и представлен на рисунке 5.

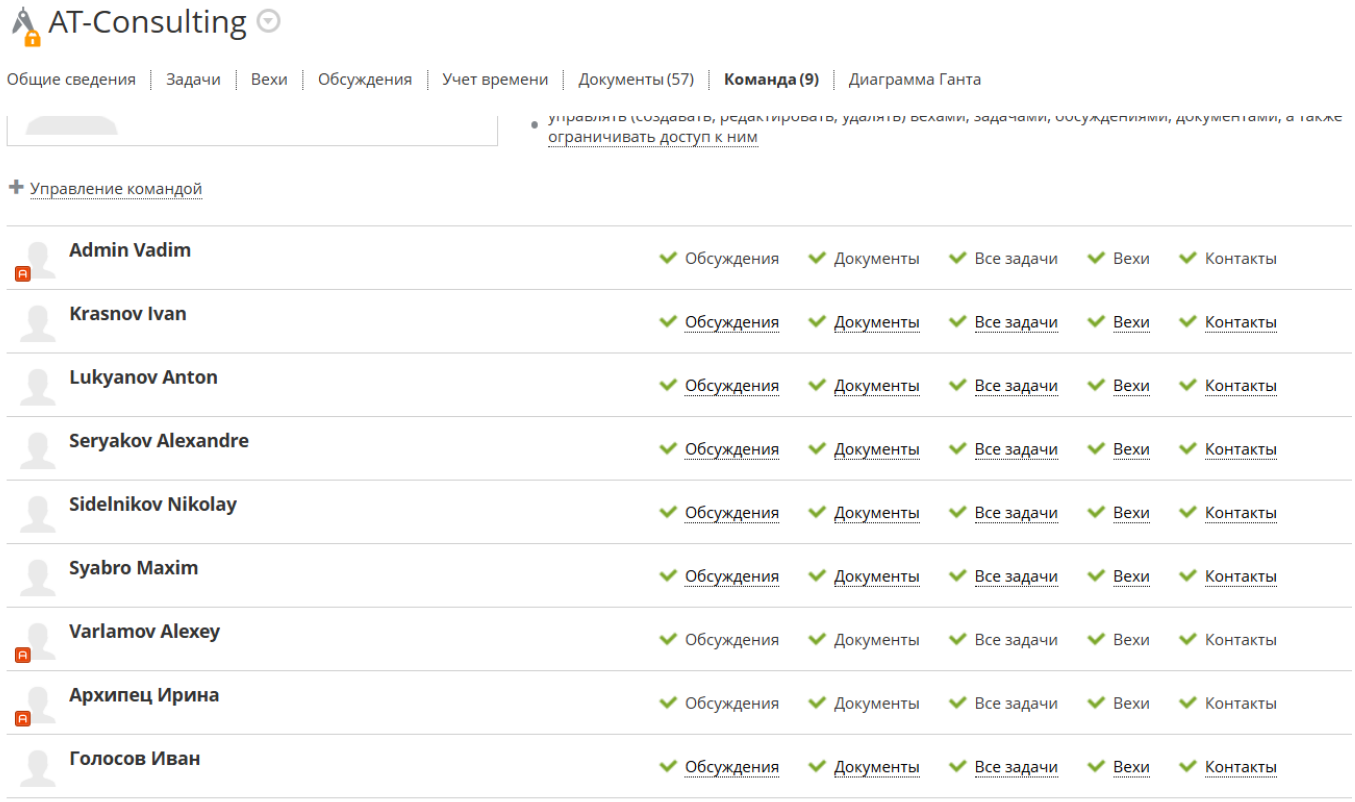

Рисунок 4 − Настройка ролей доступа к OnlyOffice

8.2. Свидетельство уникальной идентификации элементов конфигурации средства

Разработка ведется в двух постоянных ветках: тестовой и продуктовой. Дополнительно создаются временные ветки для разработки новой функциональности средства, доработки существующего функционала или исправления ошибок, которые содержат в названии номер задачи из GitLab. Пример уникальной идентификации задач в GitLab представлен на рисунке 5.

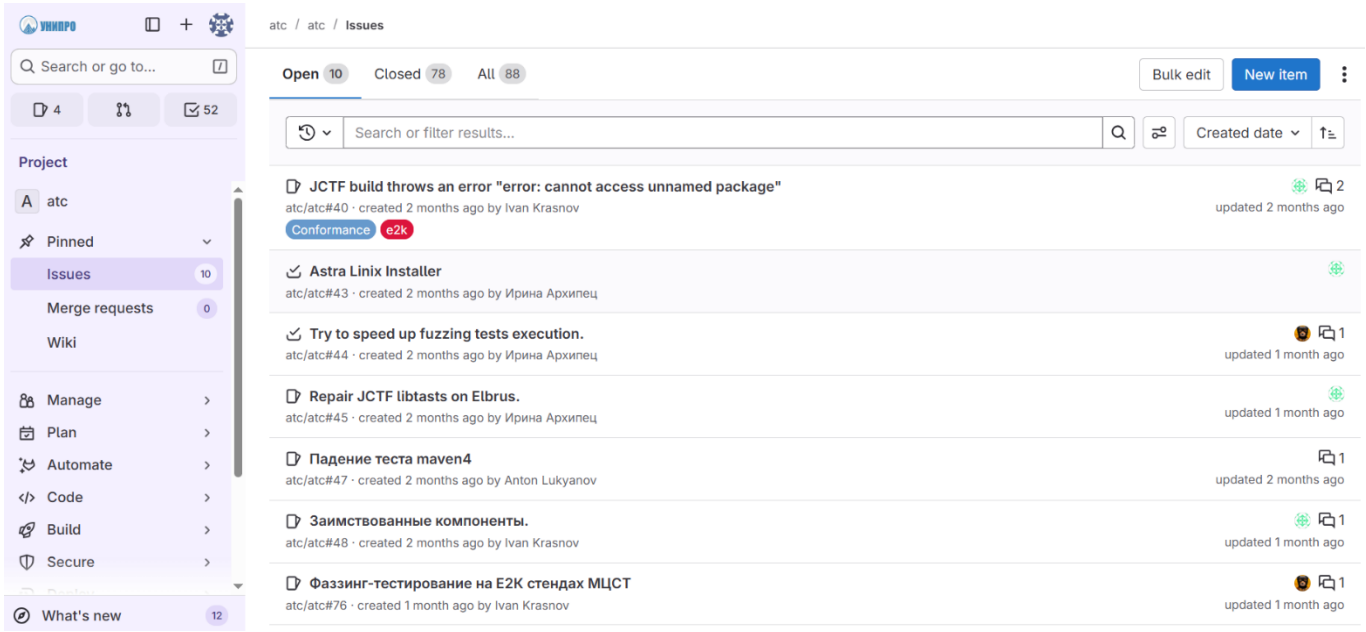

Рисунок 5 – Задачи в GitLab

На рисунке 6 представлены примеры веток в Git.

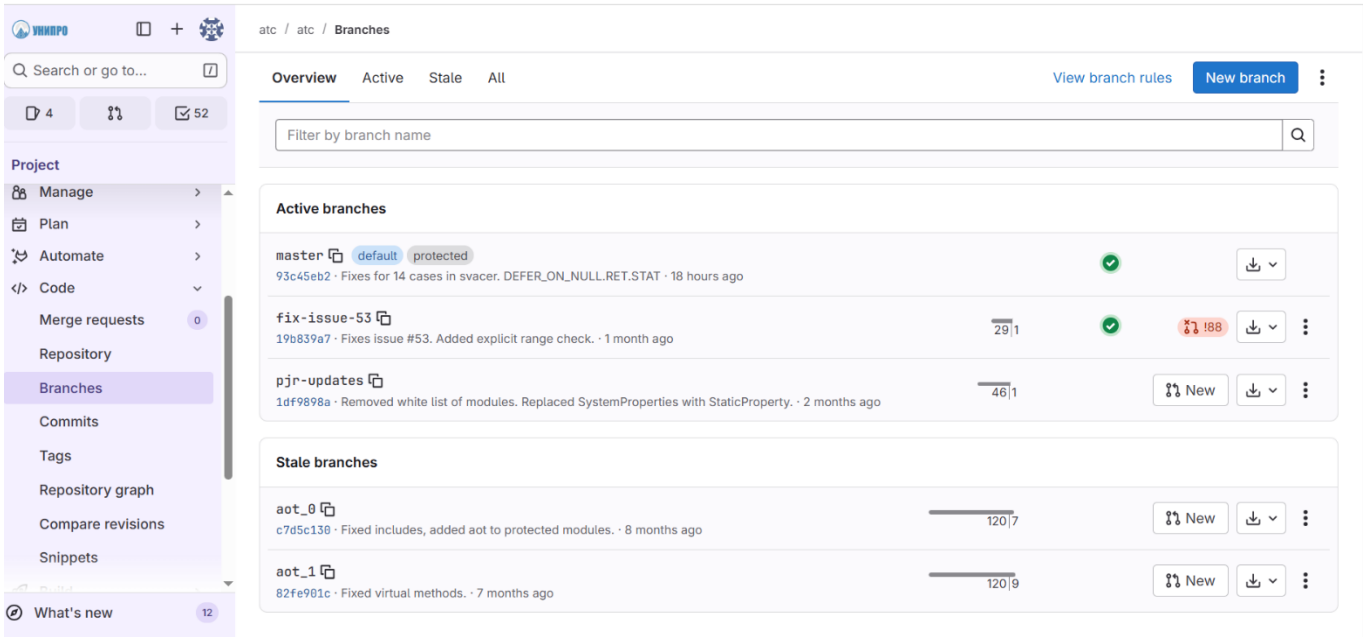

Рисунок 6 – Метка версии релиза в репозитории

Данные ветки после завершения и готовности запланированного релиза интегрируются в продуктовую ветку.

Система контроля версий программного проекта GitLab использует метки и ветки для контроля версий файлов. В ключевых точках развития проекта, таких как выпуск новой версии, создаются метки.

Интеграция временной ветки в продуктовую ветку и создание метки релиза представлены на рисунке 7.

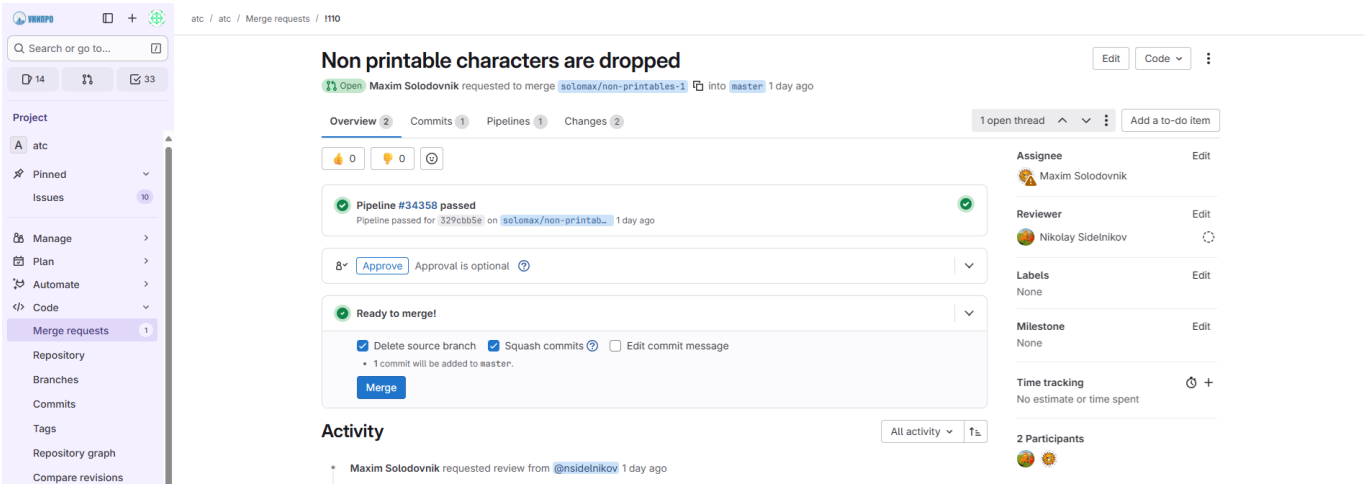

Рисунок 7 – Интеграция временной ветки в продуктовую ветку

8.3. Свидетельство операций, выполняемых в среде разработки

Все добавления и изменения в коде помещаются в хранилище с автоматически генерируемым свидетельством. Свидетельство содержит текущий идентификатор Git, дату и время, период времени, прошедший с момента последнего изменения, идентификатор пользователя, сделавшего изменение, содержание архивной записи и комментарий. Также доступен просмотр отличий от предыдущей версии (поле «Diff»).

Все обнаруженные ошибки и проблемы регистрируются в системе GitLab. После устранения ошибок регистрационные номера заносятся в строку комментариев Git (рисунок 9).

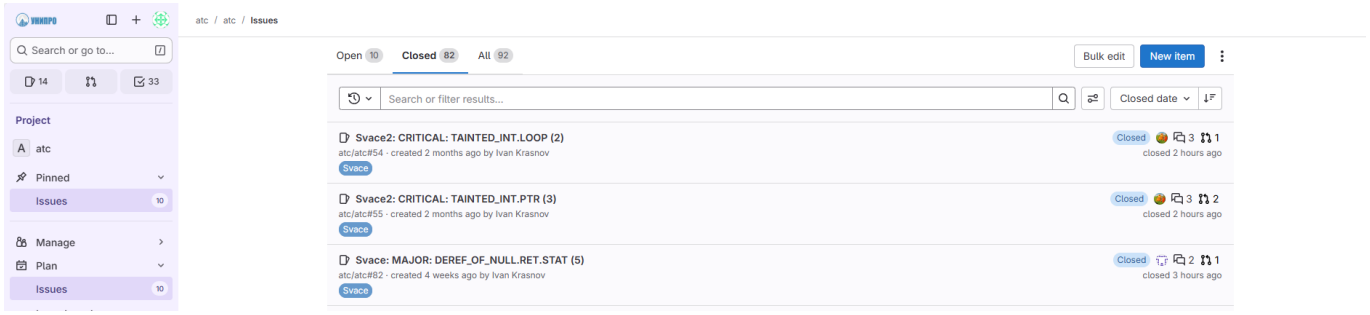

Рисунок 8 – Пример фиксации решенных задач в системе контроля версий

9. СВИДЕТЕЛЬСТВО СОБЛЮДЕНИЯ МЕР БЕЗОПАСНОСТИ

Описанные в настоящем документе меры безопасности содержатся в «Правилах внутреннего трудового распорядка», должностных инструкциях и других внутренних нормативных документах, с которыми сотрудники разработки и вспомогательный персонал ознакомлены под роспись.

Действия сотрудников разработки и вспомогательного персонала по соблюдению настоящих мер отражаются в соответствующих журналах, служебных базах данных и других средствах административного и технического контроля ООО «ЭйТи Консалтинг».

ПЕРЕЧЕНЬ ТЕРМИНОВ

Термин	Расшифровка
Astra Linux	Операционная система на базе ядра Linux, созданная для комплексной защиты информации и построения защищенных автоматизированных систем
C++	Высокоуровневый, компилируемый, статически типизированный язык программирования общего назначения
Git	Система контроля версий, которая позволяет отслеживать изменения в файлах проекта и управлять ими
GitLab	Веб-инструмент жизненного цикла DevOps с открытым исходным кодом, представляющий систему управления репозиториями кода для Git с собственной вики, системой отслеживания ошибок, CI/CD пайплайном и другими функциями
Java	Высокоуровневый, основанный на классах, объектно-ориентированный язык программирования, который разработан для минимизации зависимостей от реализации
javac	Оптимизирующий компилятор языка Java
Make	Система автоматизации сборки программного обеспечения, которая управляет процессами компиляции и линковки на основе правил, определенных в Makefile-файлах, определяя зависимости между файлами исходного кода и выполняя соответствующие команды для генерации целевых артефактов с обеспечением инкрементальной сборки измененных компонентов проекта.
Visual Studio Code	Текстовый редактор, разработанный Microsoft для Windows, Linux и macOS
Рефакторинг кода	Процесс улучшения внутренней структуры программы для лучшего понимания ее работы. Направлен на повышение читабельности кода без изменения функционала

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

Сокращение	Расшифровка
ОО	Объект оценки
ООО	Общество с ограниченной ответственностью
ОС	Операционная система
ПО	Программное обеспечение
ПО «Java AT»	Программное обеспечение «Защищенная среда разработки и исполнения «Java AT»»
УК	Управление конфигурацией
ФСТЭК России	Федеральная служба по техническому и экспортному контролю