Общество с ограниченной ответственностью   
«Современная интеграция»

**Программное обеспечение «Система автоматизации учета ИТ и ИБ услуг»**

Документация жизненного цикла

Листов 18

Москва 2023 г.

Аннотация

Данный документ представляет собой описание процедур управления конфигурацией в рамках жизненного цикла ПО, а также описание мер безопасности, применяемых при разработке программного обеспечения «Система автоматизации учета ИТ и ИБ услуг».

Оглавление

[1. Общие сведения о Системе 4](#_Toc138621155)

[2. Жизненный цикл Системы 6](#_Toc138621156)

[2.1 Модель жизненного цикла 6](#_Toc138621157)

[2.2 Стадии жизненного цикла 6](#_Toc138621158)

[2.2.1 Определение требований 7](#_Toc138621159)

[2.2.2 Проектирование 7](#_Toc138621160)

[2.2.3 Реализация 8](#_Toc138621161)

[2.2.4 Тестирование и отладка 8](#_Toc138621162)

[2.2.5 Эксплуатация и сопровождение 8](#_Toc138621163)

[3. Управление конфигурацией Системы 9](#_Toc138621164)

[3.1 Элементы конфигурации 9](#_Toc138621165)

[3.2 Дистрибутив 9](#_Toc138621166)

[3.3 Исходные тексты 9](#_Toc138621167)

[3.3.1 Порядок оформления исходного кода 9](#_Toc138621168)

[3.4 Проектная документация 10](#_Toc138621169)

[3.5 Свидетельства тестирования 10](#_Toc138621170)

[3.6 Эксплуатационная документация 11](#_Toc138621171)

[3.7 Инструментальные средства разработки 11](#_Toc138621172)

[4. План управления конфигурацией 13](#_Toc138621173)

[4.1 Контроль выполнения действий при разработке Системы 13](#_Toc138621174)

[4.2 Программное обеспечение управления конфигурацией 13](#_Toc138621175)

[4.3 Уникальная идентификация элементов конфигурации 14](#_Toc138621176)

[4.4 Резервное копирование 14](#_Toc138621177)

[5. Меры безопасности при разработке 15](#_Toc138621178)

[5.1 Физические меры 15](#_Toc138621179)

[5.2 Организационные меры 15](#_Toc138621180)

[5.3 Процедурные меры 16](#_Toc138621181)

[5.4 Технические меры 16](#_Toc138621182)

[6. Обновление программного обеспечения 17](#_Toc138621183)

# **Общие сведения о Системе**

Наименование: Программное обеспечение «Система автоматизации учета ИТ и ИБ услуг».

Разработчик: Общество с ограниченной ответственностью «Современная интеграция» (ООО «СОВИНТЕГРА»).

Адрес разработчика: 108811, г. Москва, 22-й км. Киевского шоссе (п. Московский), двлд. 4, стр. 1, блок б, этаж 9.

Программное обеспечение «Система автоматизации учета ИТ и ИБ услуг» (далее – ПО, Система) предназначено для автоматизации учета оказанных в соответствии с заключенными договорами услуг в сфере информационных технологий и информационной безопасности.

Системы представлена трехуровневой архитектурой:

1. уровень хранения данных (сервер баз данных);
2. уровень бизнес-логики (ядро Системы);
3. уровень представления данных (веб-интерфейсы взаимодействия с пользователями - возможность запуска веб-приложения Системы в браузере).

Архитектура Системы обеспечивает:

1. отказоустойчивость Системы как в целом, так и отдельных ее компонентов;
2. возможность масштабирования;
3. возможность модернизации компонентов без потери общего функционала Системы.

Система предоставляет следующие возможности:

* контроль объема потребления услуги выдачи ОТП-ключей, услуг «Интернет» и «Exchange», услуги по защите АРМ;
* создание выборочных и сводных отчетов по компаниям-контрагентам, оплачивающим оказанные услуги;
* редактирование принадлежности пользователя, которому оказаны услуга, к организации-плательщику и договору с целью решения задачи оплаты услуг;
* добавление, удаление, редактирование информации о контрагентах, в том числе о заключенных с ними договорах, используемых подсетях и т.д.;
* настройка параметров сервера Системы;
* управление тарифами на оказываемые услуги;
* управление организационной структурой заказчиков: добавление и исключение подразделений, подрядчиков, назначение кураторов и т.д.;
* аудит выполняемых в Системе действий пользователей.

Веб-сервер Системы обеспечивает хранение и обработку следующей информации:

* списки компаний-контрагентов и номера договоров с ними на оказание услуг;
* списки принадлежности пользователей к компаниям-контрагентам;
* параметры соединений с SAS, Active Directory;
* списки пользователей системы (списки операторов системы) по ролям;
* сгенерированные отчеты по объемам оказанных услуг;
* параметры приложения;
* тарифы на оказание услуг;
* события, зарегистрированные в Системе;
* договорные лимиты контрагентов по оказанию услуг.

# **Жизненный цикл Системы**

## **Модель жизненного цикла**

При разработке Системы используетсяитерационная модельжизненного цикла, предполагающая разбиение проекта на части (этапы, итерации) и прохождение этапов жизненного цикла на каждом из них. Каждый этап является законченным сам по себе, совокупность этапов формирует конечный результат – Система, готовая к передаче в эксплуатацию конечному пользователю.

На каждой итерации разработка приближается к конечному желаемому результату или уточняются требования к результату по ходу разработки, и соответственно в любой момент текущая итерация может оказаться последней или очередной на пути к завершению.

Данный подход позволяет бороться с неопределенностью, снимая ее этап за этапом, и проверять правильность технического, маркетингового или любого другого решения на ранних стадиях.

Цель каждой итерации — это получение версии ПО, включающей в себя как новые или переработанные возможности, реализованные в ходе текущей итерации, так и функциональность всех предыдущих итераций. Результат же финальной итерации содержит всю требуемую функциональность Системы.

Каждая итерация характеризуется очередной версией Системы: v.1, v.2, v.3 и т.д. Числами после «v» обозначают масштабные изменения в ядре Системы.

## **Стадии жизненного цикла**

Стадии жизненного цикла при разработке Системы следующие:

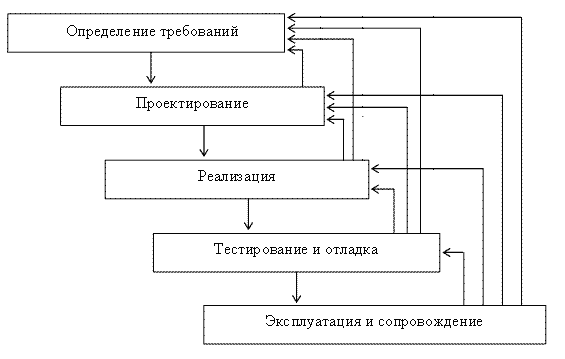


Рисунок 1 – Стадии жизненного цикла

Стрелки, ведущие вверх, обозначают возвраты к предыдущим этапам, квалифицируемые как требование повторить стадию для исправления обнаруженной ошибки. Например, переход от стадии «Эксплуатация и сопровождение» к стадии «Тестирование и отладка» подразумевает работы в рамках рекламаций, предъявляемых в ходе эксплуатации Системы. Чтобы понять, о каких ошибках идет речь в рекламации, разработчикам необходимо воспроизвести пользовательскую ситуацию у себя, т.е. выполнить действия, которые обычно относят к тестированию.

### **2.2.1 Определение требований**

Требования к Системе формируются Заказчиком и фиксируются техническим заданием на разработку. По мере необходимости в требования могут вноситься изменения по согласованию с разработчиком. Все изменения должны быть в обязательном порядке задокументированы.

### **2.2.2 Проектирование**

С учетом сформированных требований к Системе и к функциям, реализуемым с ее помощью, команда разработки выполняет проектирование архитектуры системы, а также дизайн ее графического интерфейса. На данной стадии могут уточняться требования, а также формироваться новые.

### **2.2.3 Реализация**

На основании сформированных требований и спроектированной архитектуры командой разработки ведутся работы по реализации функционала Системы. В рамках данной стадии разработчиком могут уточняться требования и архитектура Системы, вноситься изменения в состав подсистем и модулей, адаптироваться графический интерфейс под изменяющиеся требования. На данной стадии параллельно с разработкой ПО ведется разработка эксплуатационной документации для него.

### **2.2.4 Тестирование и отладка**

Каждому релизу Системы предшествует стадия ее тестирования с использованием инструментальных средств или без таковых. Тестирование выполняется командой разработки и тестировщиками. Выявленные таким образом недостатки сразу устраняются, после чего Система подвергается всем необходимым видам тестирования до получения полностью работоспособного программного обеспечения.

### **2.2.5 Эксплуатация и сопровождение**

Стабильно работающее полнофункциональное ПО передается в эксплуатацию Заказчику на основании соответствующих договорных отношений. В ходе эксплуатации Системы оказываются услуги по ее сопровождению, включающие консультирование пользователей по вопросам функционала, а также по устранению недостатков и уязвимостей Системы. За сопровождение Системы отвечают сотрудники из команды разработки.

# **Управление конфигурацией Системы**

## **Элементы конфигурации**

Система управления конфигурацией отслеживает следующие элементы конфигурации:

* Дистрибутив;
* Исходные тексты;
* Проектную документацию;
* Свидетельства тестирования;
* Эксплуатационную документацию;
* Инструментальные средства разработки.

## **Дистрибутив**

Дистрибутив Системы, соответствующий всем предъявленным требованиям и переданный в эксплуатацию конечному пользователю, имеет версию 2.0.23.0. Эталонная версия дистрибутива хранится у ведущего разработчика с фиксацией контрольной суммы, обеспечивающей подтверждение целостности дистрибутива.

## **Исходные тексты**

Исходные тексты, соответствующие итоговому дистрибутиву, а также всем промежуточным релизам, хранятся в системе управления Git, с обязательной идентификацией версий.

### **3.3.1 Порядок оформления исходного кода**

Порядок оформления исходного кода строится так, чтобы за счет определенного визуального оформления элементов программы повысить информативность кода для человека.

Стандарт оформления кода определяется разработчиком для каждого инструментального средства, фиксируется в системе Git и описывает:

* способы выбора названий и используемый регистр символов для [имен переменных](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B5%D0%BC%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) и других [идентификаторов](https://ru.wikipedia.org/wiki/%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%82%D0%BE%D1%80): запись типа переменной в ее идентификаторе и регистр символов, [использование](https://ru.wikipedia.org/wiki/Snake_case) [знаков подчеркивания](https://ru.wikipedia.org/wiki/%D0%97%D0%BD%D0%B0%D0%BA_%D0%BF%D0%BE%D0%B4%D1%87%D1%91%D1%80%D0%BA%D0%B8%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) для разделения слов;
* стиль [отступов](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D1%81%D1%82%D1%83%D0%BF_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) при оформлении [логических блоков](https://ru.wikipedia.org/wiki/%D0%91%D0%BB%D0%BE%D0%BA_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)): используются ли [символы табуляции](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BC%D0%B2%D0%BE%D0%BB_%D1%82%D0%B0%D0%B1%D1%83%D0%BB%D1%8F%D1%86%D0%B8%D0%B8), ширина отступа;
* использование [пробелов](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B1%D0%B5%D0%BB) при оформлении логических и арифметических выражений;
* правила использования указателей;
* правила именования пакетов и файлов;
* правила логирования;
* правила обработки ошибок.

Вне стандарта подразумевается:

* отсутствие [магических чисел](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D0%B3%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%BE%D0%B5_%D1%87%D0%B8%D1%81%D0%BB%D0%BE_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5));
* ограничение размера кода по горизонтали (чтобы помещался на экране) и вертикали (чтобы весь код файла держался в памяти), а также функции или метода в размер одного экрана.

## **Проектная документация**

Все материалы, сопутствующие стадии проектирования Системы, подлежат хранению. К таким материалам относятся эскизный и технический проекты, демонстрирующие разбиение Системы на подсистемы и модули, архитектура, описывающая меры по обеспечению безопасности обрабатываемой информации и осуществляемых взаимодействий, спецификация, демонстрирующая внешний вид интерфейсов разрабатываемой Системы и т.д. В случае внесения изменений в проектные материалы, хранению подлежат и все предыдущие варианты.

## **Свидетельства тестирования**

На стадии тестирования и отладки выполняются следующие виды тестирования Системы:

* Экспертиза исходного кода;
* Статический анализ кода;
* Динамический анализа кода, включая фаззинг-тестирование;
* Тестирование на проникновение;
* Функциональное тестирование.

Периодичность, используемые инструментальные средства и объемы выполняемых тестов определяются разработчиком, исходя из степень готовности промежуточного релиза или же Системы в целом, из результатов предшествующих тестов, а также из специфики тестируемых функциональных возможностей.

Тестирование выполняется специалистами команды разработки и тестировщиками.

Результаты прохождения тестов фиксируются с использованием систем хранения исходных текстов и учета проектов.

## **Эксплуатационная документация**

По мере разработки Системы создаются следующие эксплуатационные документы, предоставляемые в дальнейшем конечному пользователю:

* Руководство пользователя;
* Руководство администратора.

По мере внесения изменений в функции Системы эксплуатационная документация актуализируется.

## **Инструментальные средства разработки**

Система разработана с использованием следующих языков программирования:

* c# (описание доступно по ссылке <https://learn.microsoft.com/ru-ru/dotnet/csharp/>);
* SQL (описание доступно по ссылке <https://sql-academy.org/ru>);
* Javascript (описание доступно по ссылке <https://learn.javascript.ru/>);
* html5 (описание доступно по ссылке <http://htmlbook.ru/html5>);
* css (описание доступно по ссылке <https://www.w3schools.com/css/>).

Для разработки Системы используются инструменты, приведенные в таблице ниже.

Таблица 1 – Инструментальные средства разработки

| Система | Продукт | Версия | Разработчик | Ссылка на документацию |
| --- | --- | --- | --- | --- |
| Система управления проектом | Redmine | 5.0 | Некоммерческий проект с открытым исходным кодом | <https://www.redmine.org/projects/redmine/wiki/Guide> |
| Система учета запросов и ошибок |
| Система хранения исходных текстов | GitLab Community Edition | 15.0 | Некоммерческий проект с открытым исходным кодом | <https://git-scm.com/doc> |
| Средства сборки | Maven | 4.0.0 | Некоммерческий проект с открытым исходным кодом | <https://maven.apache.org/> |
| Средства тестирования | TestLink | 1.9.19 | Некоммерческий проект с открытым исходным кодом | <https://github.com/TestLinkOpenSourceTRMS/testlink-documentation> |
| Виртуальный стенд | Proxmox | 7.4 | Некоммерческий проект с открытым исходным кодом | <https://www.proxmox.com/en/downloads> |

# **План управления конфигурацией**

## **Контроль выполнения действий при разработке Системы**

Система управления конфигурацией контролирует все операции, выполняемые в отношении элементов конфигурации Системы на всех этапах их жизненного цикла, такие как создание элемента конфигурации, чтение, запись, изменение. Права сотрудников на выполнение определенных операций предусмотрены их должностными обязанностями.

Операции над элементами конфигурации выполняются на автоматизированных рабочих местах сотрудников команд разработки и тестирования. АРМ работают под управлением операционных систем семейства Linux и Windows. Каждый сотрудник работает под именной учетной записью, позволяющей однозначно идентифицировать ее владельца. Журналирование событий, генерируемых действиями пользователей на своих АРМ, осуществляется средствами управляющих операционных систем. Журналирование событий, генерируемых в системах управления проектами, централизованного хранения данных, а также системами хранения исходного кода, сборки и тестирования, выполняется средствами этих систем. В обоих случаях фиксируется следующая информация: учетное имя инициатора события, время, операция, результат. Доступ к просмотру журнала событий имеют администраторы систем, в случае ведения локального журнала – пользователь АРМ, обладающий правами администратора.

## **Программное обеспечение управления конфигурацией**

Управление конфигурацией осуществляется с использованием следующих систем:

* система хранения исходных текстов и управления версиями, сборки и тестирования продуктов Git фиксирует изменения в исходном коде с указанием даты, времени и имени пользователя, внесшего правки, позволяет подтверждать или отменять внесенные правки, выбирать необходимую версию файлов, с учетом изменений, локально изменять и применять изменения к требуемым файлам. Также система позволяет обеспечить управление доступом к репозиторию с исходными текстами проекта на основе учетных записей. Сборка проекта происходит в автоматизированном режиме по запросу пользователя либо периодически в соответствии с регламентом сборки;
* система планирования работ и управленческого учета, учета запросов и ошибок Redmine 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.

## **Уникальная идентификация элементов конфигурации**

Git позволяет контролировать состав файлов исходных текстов и других исходных модулей, из которых собирается Сситема, и обеспечивать их уникальную поверсионную идентификацию. Система уникальной идентификации элементов конфигурации строится на основе меток – дополнительного механизма Git, расширяющего штатные возможности нумерации версий. Метка представляет собой буквенное имя версии, которое присваивается как дополнительный идентификатор зафиксированной версии образа программного обеспечения.

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

## **Резервное копирование**

Резервное копирование выполняется администраторами в отношении следующих информационных систем:

* система управленческого учета Redmine - резервное копирование выполняется раз в неделю;
* система хранения исходных текстов и управления версиями Git - резервное копирование выполняется три раза в неделю.

Хранению подлежат последние пять резервных копий системы хранения исходных текстов и три копии системы управленческого учета.

Резервное копирование выполняется средствами системы Proxmox. Этим же инструментом выполняется и восстановление данных администратором соответствующей системы из резервной копии в случае возникновения такой необходимости в срок, не превышающий 24 часов после обнаружения недоступности текущей конфигурации системы.

# **Меры безопасности при разработке**

Для обеспечения безопасности элементов конфигурации в инфраструктуре среды разработки программного обеспечения применяются физические, организационные, процедурные и технические меры.

## **Физические меры**

При размещении технических средств (далее – ТС) разработки программного обеспечения выполняются следующие условия:

* размещение, охрана и специальное оборудование помещений, в которых установлены ТС разработки и ведется работа с носителями персональной ключевой информации, исключает возможность бесконтрольного проникновения в них посторонних лиц, прослушивания ведущихся там переговоров и просмотра помещений посторонними лицами, а также гарантирует сохранность находящихся в этих помещениях конфиденциальных документов;
* входные двери помещений оборудованы внутренними замками, гарантирующими надежное закрытие дверей при выходе из помещения и в нерабочее время. Окна и двери оборудованы охранной сигнализацией, связанной с центральным пультом наблюдения за сигнализацией поста охраны;
* служебные помещения, в которых находятся ТС разработки, средства, используемые для хранения бумажных и цифровых материалов, оборудуются средствами вентиляции и кондиционирования воздуха, обеспечивающими соблюдение установленных параметров температурно-влажностного режима, вентиляции и очистки воздуха;
* по окончании рабочего дня помещения закрываются и сдаются под охрану.

## **Организационные меры**

Организационные меры включают следующие:

* на случай пожара, аварии или стихийного бедствия разработаны специальные инструкции, в которых предусматривается порядок вызова администрации, должностных лиц, вскрытие помещений, очередность и порядок эвакуации ТС, документов и носителей данных и дальнейшего их хранения;
* уборка помещения, обслуживание оборудования систем жизнеобеспечения осуществляется назначенным персоналом при выключенных мониторах в присутствии должностных лиц;
* выполняется ознакомление и обучение сотрудников правилам работы с системами, обеспечивающими процессы управления конфигурацией;
* принимаются меры по надежному сохранению в тайне паролей доступа, ключевых дистрибутивов и другой ключевой информации, размещенной на съемных носителях;
* проводится периодическое обучение сотрудников команд разработки и тестирования принципам безопасной разработки программного обеспечения, а также использованию специализированных инструментальных средств.

## **Процедурные меры**

В качестве процедурных мер при обеспечении безопасности элементов конфигурации выполняются следующие:

* предоставление доступа к среде разработки или к конкретным объектам среды, таким как оборудование разработки, согласно должностным обязанностям и указаниям руководства;
* незамедлительная отмена прав доступа к среде разработки лиц при их исключении из состава команды разработки или увольнении;
* запрет неконтролируемой передачи защищаемых материалов из среды разработки;
* запрет бесконтрольного пребывания посторонних лиц в помещениях, где расположены технические средства среды разработки;
* распределение ролей и обязанностей по обеспечению непрерывного применения мер безопасности и обнаружения нарушений безопасности.

## **Технические меры**

Среди технических мер безопасности в отношении элементов конфигурации выполняются следующие:

* антивирусный контроль АРМ разработчиков и серверов применяемых систем управления, хранения и учета;
* контроль доступа к элементам конфигурации;
* периодическое выполнение резервного копирования информации, содержащейся в электронных системах хранения и учета.

# **Обновление программного обеспечения**

Для ПО предусмотрен выпуск следующих типов обновлений:

* + - обновления, направленные на устранение уязвимостей и недостатков;
    - обновления, направленные на добавление функции (функций), на совершенствование реализации функции (функций).

Для устранения уязвимостей разработчик периодически проводит поиск известных (подтвержденных) уязвимостей в общедоступных источниках информации об уязвимостях. Также выполняется прием сообщений об уязвимостях от эксплуатирующих Систему организаций с использованием телефонной связи и электронной почты.

При выявлении или получении информации об уявимости разработчик проводит ее анализ, учитывая следующие критерии:

1) тип ошибки;

2) версия программного обеспечения, подверженная уязвимости;

3) уровни опасности уязвимости:

* критическая (Critical);
* высокая (High);
* средняя (Medium);
* низкая (Low).

4) информация об устранении.

На основании проведенного анализа разработчик выпускает новую версию ПО с устраненной уязвимостью или реализует ограничения по применению Системы, нейтрализующие уязвимость либо делающие невозможным ее эксплуатацию злоумышленниками. Данные мероприятия должны быть вполнены в максимально короткий срок. Ограничения по применению отражаются в эксплуатационной документации.

Информация о недостатках системы, выявленных эксплуатирующими организациями, принимается в форме рекламаций, направленных официальным письмом.

Недостатки, выявленные разработчиком в ходе тестирования, фиксируются в соответствующих системах учета и ставятся в план по устранению на ближайший релиз Системы.

Добавление новых функций или совершенствование реализации имеющихся функций выполняются по согласованию с эксплуатирующими организациями путем выхода новой версии Системы, прошедшей обязательные процедуры тестирования.

Любые обновления Системы доводятся до сведения эксплуатарующих организаций средствами электронной почты.