Политика поддержки и жизненного цикла системы Web-SCADA

Версия 1.0
Дата составления: 22.07.2024

1. Цель и область действия

Настоящий документ определяет процессы, роли и процедуры, обеспечивающие стабильную эксплуатацию, своевременное устранение неисправностей и эволюционное развитие программного комплекса Web-SCADA на протяжении всего его жизненного цикла.

Область действия: Документ распространяется на все этапы жизненного цикла системы после ее сдачи в промышленную эксплуатацию: штатная работа, инциденты, изменения, обновления, вывод из эксплуатации.

2. Организационная модель поддержки

2.1. Роли и обязанности (со стороны Исполнителя - компании-разработчика)

Поддержка осуществляется по трехуровневой модели:

Уровень	Роль	Ключевые обязанности	Квалификационные требования
Уровень 1 (L1)	Инженер поддержки	Первичный прием и классификация обращений. Базовая диагностика (админка, доступность узлов). Конфигурация через интерфейсы системы. Эскалация сложных проблем на L2.	Опыт администрирования Linux, умение работать с логами, проверять состояние сервисов, понимание базовых принципов работы сети.
Уровень 2 (L1)	Разработчик-инженер	Глубокий анализ логов и кода. Исправление ошибок (багов) в ПО. Настройка сложных интеграций и промышленных протоколов. Решение проблем, эскалированных с L1.	Знание Python, Django, Redis, PostgreSQL. Опыт работы с промышленными протоколами (Modbus, OPC UA). Глубокое понимание сетевых взаимодействий и архитектуры системы.
Уровень 3 (L1)	Ведущий разработчик / Архитектор	Анализ архитектурных проблем. Принятие решений по сложным изменениям и оптимизациям. Разработка критичных исправлений и новых значимых функций.	Экспертное знание всей кодовой базы и архитектуры системы. Опыт проектирования распределенных систем.

2.2. Роли и обязанности (со стороны Заказчика)

Для эффективной поддержки Заказчик назначает Ответственного представителя, который:

- Обеспечивает физический доступ к оборудованию и узлам системы.

- Решает вопросы на уровне локальной сети объекта (проверка кабелей, коммутаторов, питания).
- Является единой точкой контакта для оперативной коммуникации.
- Принимает решение о согласовании выезда специалиста на объект согласно договору.

2.3. Коммуникационные каналы и передача проекта

- Основные каналы связи для инцидентов: Выделенный Telegram-чат проекта и телефон для экстренных случаев.
- Передача проекта в поддержку: По завершении внедрения команда разработки передает команде поддержки пакет документов, включающий:
    1. Архитектурное описание.
    2. Руководство администратора и пользователя.
    3. Список оборудования с деталями взаимодействия (IP-адреса, протоколы, точки доступа).
    4. Контакты ответственных лиц с обеих сторон.

3. Процесс управления инцидентами

Инцидент – любое отклонение от штатной работы системы, влияющее на ее функциональность или доступность.

3.1. Шаги обработки инцидента:

1. Регистрация и классификация: Инженер L1 принимает обращение от Заказчика через Telegram/телефон. Фиксирует описание проблемы, время возникновения, объект и узел.
2. Первичная диагностика (L1): Инженер проводит анализ по чек-листу:
    - Проверяет состояние системы через админ-панель Django.
    - Проверяет доступность проблемного контроллера/оборудования (ping, статус соединения в системе).
    - Анализирует логи сервера (Nginx, Gunicorn, Django).
    - При необходимости запрашивает доступ для анализа логов на полевом контроллере (логи хранятся локально на каждом устройстве).
3. Решение и эскалация:
    - Если проблема решается настройкой или перезапуском сервиса – инженер L1 выполняет действие и согласовывает его с Заказчиком.
    - Если выявлена программная ошибка или сложная проблема с интеграцией – инженер L1 эскалирует инцидент разработчику L2, передавая все собранные данные и логи.
    - Разработчик L2 проводит углубленный анализ, исправляет код и готовит обновление.
4. Закрытие: После устранения причины инцидента ответственный инженер информирует Заказчика о решении и закрывает инцидент.

3.2. Сценарий эскалации "Нет данных с контроллера":

1. L1: Проверка статуса контроллера на сервере. Если связь отсутствует – взаимодействие с Ответственным Заказчика для проверки сети и питания. Если связь есть, но данные некорректны – анализ логов контроллера.
2. L2: Если в логах контроллера обнаружена ошибка обработки протокола или внутренний сбой – разработчик L2 анализирует код соответствующего микросервиса, исправляет баг, готовит патч.
3. Выезд на объект: Инициируется по согласованию с Заказчиком в рамках договора сопровождения, если проблема не может быть решена удаленно (например, требуется замена или ремонт аппаратной части).

4. Процесс управления изменениями и обновлениями

Все изменения в промышленной среде производятся строго в соответствии с договором на техническое сопровождение.

4.1. Внесение изменений и обновление ПО:

- Серверная часть: Обновление выполняется путем выкатки последней версии кода из Git-репозитория. Обязательным этапом является предварительное тестирование обновления в тестовой среде, максимально приближенной к боевой (если таковая существует).
- Полевые контроллеры: Обновление производится индивидуально для каждого контроллера. Процедура выполняется удаленно или, при необходимости, с физическим доступом.
- План отката: Для каждого обновления должен быть определен и согласован план отката на предыдущую стабильную версию (например, восстановление из бекапа конфигурации или переключение на резервный инстанс).

4.2. Управление требованиями:

Новые функциональные требования и доработки от Заказчика собираются, ранжируются индивидуально по согласованию и вносятся в бэклог разработки. Приоритизация осуществляется совместно, исходя из бизнес-ценности и технической сложности.

5. Процессы обеспечения стабильности

5.1. Резервное копирование и восстановление:

- Что бэкапируется: База данных PostgreSQL, конфигурационные файлы системы, медиафайлы, критичные логи.
- Политика хранения: Частота и глубина хранения бекапов определяется индивидуально для каждого объекта, исходя из критичности данных, технологических процессов и доступного объема хранилища.
- Хранилище: Бекапы хранятся на выделенном диске (в т.ч. в RAID-массиве) или на сетевом хранилище (NAS/SAN) в контуре Заказчика.
- Проверка восстановления: Регулярность проверки целостности и возможности восстановления из бекапа определяется критичностью системы для Заказчика и фиксируется в договоре сопровождения.

5.2. Мониторинг работоспособности:

Критически важные метрики для предупреждения инцидентов:

- Инфраструктурные: Загрузка CPU и памяти сервера, свободное место на дисках.
- Сервисные: Статус и доступность ключевых процессов (Django, Gunicorn, Nginx, PostgreSQL, Redis).
- Бизнес-логические: Количество активных подключений от полевых контроллеров, время отклика базы данных, наличие данных с ключевых тегов.

6. Управление знаниями и документацией

- База знаний: Внутренняя Wiki компании используется для накопления и структурирования информации: типовые решения, инструкции по диагностике, архитектурные решения.
- Обучение персонала Заказчика: Проводится на основе Руководства пользователя и Руководства администратора. Формат – очное или дистанционное обучение с разбором реальных сценариев работы. Дальнейшие консультации осуществляются в рабочем Telegram-чате.

7. Жизненный цикл системы

1. Внедрение: Разработка и настройка под конкретный объект. Передача в промышленную эксплуатацию.
2. Эксплуатация и поддержка: Штатная работа, управление инцидентами и изменениями в соответствии с настоящей политикой.
3. Развитие: Плановое совершенствование, добавление нового функционала, интеграция с новыми системами.
4. Вывод из эксплуатации: Архивация данных, отключение сервисов, утилизация или передача компонентов. Осуществляется по отдельному плану в конце жизненного цикла.