РЕФЕРАТ

Отчёт 16 с., 1 ч., рис., табл., источников, прил.

Объект разработки — программный компонент, управляющий охранной сигнализацией.

Цель работы — разработка программного модуля для управления устройством охранной сигнализации в программном комплексе интеграции оборудования.

Программный модуль разрабатывается исходя из: протокола взаимодействия с устройством, протокола взаимодействия с системой сбора и обработки информации, интерфейсов программного ядра системы.

В результате работы разработан программный модуль и сценарий модульного тестирования.

Программный модуль и будет интегрирован в систему охранной сигнализации.

Содержание

1 Определения

В настоящем отчете об УИР применяют следующие термины с соответствующими определениями :

Ant-Engine — утилита для автоматизации процесса подготовки к сборке *проекта* (Например, разрешение зависимостей проекта от библиотек);

Артефакты — набор файлов, которые не компилируется в рамках текущего *проекта*, но необходимы для работы программы (Например, библиотеки предназначенные для конкретной платформы (динамически линкуемые, с расширением .dll, .so) или файлы данных *(конфигурационные*));

Библиотечный файл — скомпилированные исходные коды (файлы с расширением .jar), не подлежащие компиляции при сборке *проекта*, реализующие функционал, используемый в нескольких *проектах;*

Динамический параметр — *шаблонный параметр*, который в процессе конфигурирования заполняется значением, зависящим от конфигурируемого компонента, что позволяет для разных приложений использовать один и тот же шаблон, но в результате инсталляции будут получены *конфигурационные файлы* с одинаковой структурой, но с различными значениями параметров;

Дистрибутив — это файл, содержащий в себе исполняемую программу, готовую к установке на компьютер, управляющий системой;

Инструментальный компьютер — предназначен для выполнения процедур компиляции программного обеспечения и анализа исходного кода;

Контекст комплекса — пространство параметров комплекса, сохраняемых в едином *файле-реестре;*

Конфигурационные файлы — текстовые файлы, используемые приложениями для хранения конфигурационных параметров;

Модуль — набор исходных кодов *проекта*, который может компилироваться отдельно;

Платформа — множество аппаратно-программных конфигураций;

Проект — совокупность файлов с исходным кодом, набором *артефактов* и

файлами для среды разработки, с помощью которых возможно получение всех необходимых выходных файлов (исполняемые, конфигурационные, справочные, установочные, ресурсные, библиотечные и т. д.) программного компонента или его части;

Рабочая копия — локальное дерево папок, содержащее проект, извлеченный из *SVN;*

Релиз — выходная структура *проекта*, содержащая все необходимые файлы (исполняемые, конфигурационные, справочные, установочные, ресурсные, библиотечные и т. д.), а так же наборы дополнительных программ и служебных утилит, необходимых для сборки на их основе дистрибутива и/или установки этих файлов на компьютере пользователя и последующей эксплуатации;

Файл справки — это файл, содержащий информацию о том, как пользоваться программой;

Файл-реестр (или реестр) — особый XML-документ, который содержит все значения всех *шаблонных параметров*, описанных в *шаблонных файлах;*

Файл-реестр по умолчанию — *файл-реестр*, содержащий заводские значения параметров;

Шаблонные файлы (или шаблоны) — текстовые файлы, являющиеся "исходными" для создания на их основе *конфигурационных файлов*, скриптов, и прочих видов текстовых документов, путем замены содержащихся в этих исходных файлах специальным образом сформированных параметров на значения, хранящиеся в *файле-реестре;*

Шаблонный параметр — особым образом заданный подстановочный параметр, содержащийся в шаблонном файле, чьё значение определено в *файле-реестре;*

2 Обозначения и сокращения

SVN — централизованная система управления версиями файлов;

АПИ — аппаратно-программный интерфейс;

АРМ — автоматизированное рабочее место;

ИК СФЗ — интегрированный комплекс средств и систем физической защиты;

КИТСФЗ — комплекс инженерно-технических средств и систем физической защиты;

КНЦ — концентратор датчиков;

КСА — комплекс средств автоматизации;

ЛВС — локальная вычислительная сеть;

ПУ — процессор управления;

СКУД — система контроля и управления доступом;

СОС — система охранной сигнализации;

СПО — специальное программное обеспечение;

ССОИ — система сбора и обработки информации;

СУДОС — система управления доступом и охранной сигнализацией;

ТС — техническое средство;

JDK — Java development kit (комплект разработки на Java)

JVM — Java virtual machine (виртуальная машина Java)

3 Введение

Данная учебно-исследовательская работа заключается в разработке программного модуля, управляющего устройством в охранной системе. Разработка ведется исходя из:

- Протокола взаимодействия с устройством;
- Протокола взаимодействия с ССОИ;
- АПИ.

Протокол взаимодействия с устройством в данной системе унифицирует взаимодействие с различными типами охранных датчиков. Компонент разрабатывается с учётом интеграции в программный комплекс АПИ и взаимодействия с ССОИ.

4 Основная часть

4.1 Обзор работ связанных с УИР

4.1.1 Унифицированный протокол взаимодействия с ССОИ

Данный протокол необходим для унификации подключения контроллеров технических средств ИК СФЗ различных производителей к ССОИ КИТСФЗ.[1]

Аппаратно-программные интерфейсы

АПИ используются для подключения контроллеров технических средств СФЗ к ССОИ. АПИ имеет в своем составе аппаратный модуль, реализующий физический интерфейс подключения контроллера (RS-232, RS-485, RS-422, CAN и т.п.) и СПО АПИ. СПО АПИ должно[1]:

- обеспечивать обмен с ССОИ через ЛВС по унифицированному протоколу верхнего уровня;
- обеспечивать обмен информацией с контроллерами ТС через аппаратный интерфейс по специализированному протоколу ТС.

Платформой для установки АПИ являются ПУ, входящие в состав ССОИ. В ССОИ процессоры управления включены в состав КСА пунктов управления и участков контроля. ПУ представляет собой необслуживаемый компьютер с круглосуточным режимом работы. В ПУ устанавливаются (или подключаются) аппаратные модули АПИ, и на ПУ устанавливается СПО АПИ. На одном ПУ могут функционировать несколько АПИ. Обобщенная структурная схема АПИ приведена на рисунке 1.[1]

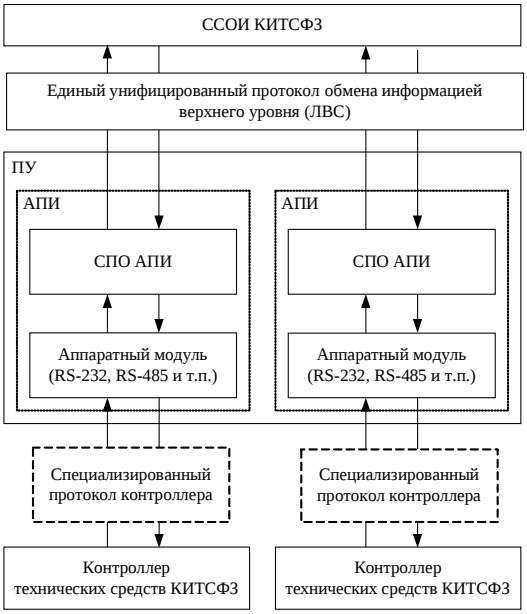

Рисунок 1- Обобщенная структурная схема АПИ

Понятие модели типа устройства

Для унификации взаимодействия с различным оборудованием в едином унифицированном протоколе обмена информацией используется понятие модели типа устройства. Модель типа устройства — это совокупность информации (характеристик) одного типа устройств (прибора, контроллера, шлейфа сигнализации, видеокамеры и т.п.). Между моделями поддерживается отношения родитель-потомок. Схема модели представлена на рисунке 2.

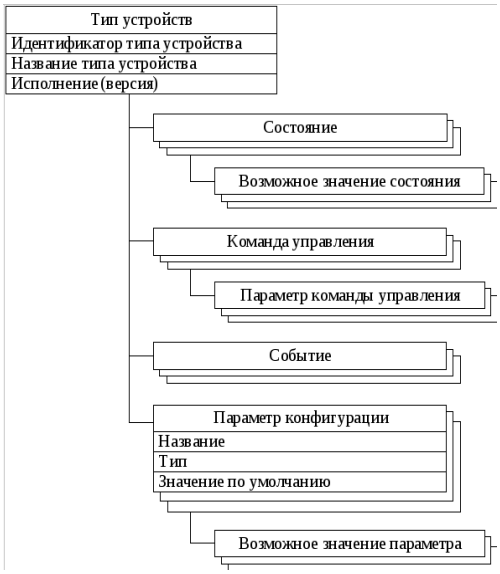

Рисунок 2- Схема модели типа устройства

Модель типа устройства формируется производителем оборудования и передается в ССОИ по единому унифицированному протоколу верхнего уровня. Модель заносится в БД ССОИ, и в дальнейшем вся работа ССОИ с оборудованием строится на основе информации о модели устройства. АПИ является устройством, у которого есть дочерние типы устройств (например, порты RS-485), у которых в свою очередь есть свои дочерние типы устройств (например, контроллеры) и т. д.[1]

Варианты подключения технических средств СФЗ к ССОИ

ТС СФЗ должны подключаться к ССОИ по одному из следующих вариантов[1]:

- Если контроллер ТС имеет возможность подключения в локальную вычислительную сеть по транспортному протоколу TCP/IP, а программное обеспечение контроллер ТС реализует единый унифицированный протокол обмена информацией верхнего уровня, то контроллер ТС подключается напрямую в ЛВС ССОИ КИТСФЗ. В этом случае программное обеспечение ТС реализует функции АПИ. Модель оборудования при этом варианте формируется в контроллере ТС и передается в ССОИ по единому унифицированному протоколу обмена информацией верхнего уровня;
- Если ТС имеет возможность подключения по стандартным аппаратным интерфейсам (RS-232, RS-485, CAN и т.п.), то производителем контроллера ТС поставляется СПО АПИ, обеспечивающее обмен информацией с контроллером ТС через аппаратный интерфейс по специализированному протоколу с одной стороны и обмен с ССОИ через ЛВС по унифицированному протоколу верхнего уровня с другой стороны. СПО АПИ и стандартный аппаратный интерфейс устанавливается на ПУ, контроллер ТС подключается к стандартному аппаратному интерфейсу. Модель оборудования в этом случае формируется СПО АПИ и переда-

ется в ССОИ по единому унифицированному протоколу обмена информацией верхнего уровня;

- Если контроллер ТС подключается по уникальному аппаратному интерфейсу, то производителем контроллера ТС поставляется аппаратный интерфейсный модуль для подключения ТС к ПУ. Также, производителем контроллера ТС поставляются СПО АПИ, обеспечивающее обмен информацией с контроллером ТС через аппаратный интерфейс по специализированному протоколу с одной стороны и обмен с ССОИ через ЛВС по унифицированному протоколу верхнего уровня с другой стороны. СПО АПИ и аппаратный интерфейс устанавливается на ПУ, контроллер ТС подключается к аппаратному интерфейсу. Модель оборудования в этом случае формируется СПО АПИ и передается в ССОИ по единому унифицированному протоколу обмена информацией верхнего уровня.

4.1.2 Протокол взаимодействия с КНЦ

4.2 Обзор инструментальных средств

4.2.1 Язык программирования Java

Для разработки программного модуля используется язык программирования Java. Java — объектно-ориентированный язык программирования, разработанный компанией Sun Microsystems (в последующем приобретённой компанией Oracle). Приложения Java обычно компилируются в специальный байт-код, поэтому они могут работать на любой виртуальной Java-машине (JVM) вне зависимости от компьютерной архитектуры[2]. Выделим существенные преимущества Java:

- Автоматическое управление сборкой мусора;
- Кроссплатформенность;

- Набор стандартных классов-коллекций (строки, массивы, карты);
- Встроенные в язык средства создания многопоточных программ.

Основные недостатки Java:

- Высокое потребление памяти, по-сравнению с аналогичной программой на C++;
- Низкая скорость работы приложений.

В текущей работе используется JDK версии 1.6.0u13. При разработке используется интегрированная среда разработки IntelliJ IDEA. IDEA предлагает удобные средства для:

- Рефакторинга исходного кода;
- Работы с системами контроля версий;
- Удобной логической организации Java приложений;
- Удобного запуска тестовых сценариев.

4.2.2 Библиотека для модульного тестирования Junit

JUnit — библиотека для модульного тестирования Java, которая предоставляет возможности удобной и гибкой организации тестовых сценариев. Модульное тестирование позволяет тестировать компоненты отдельно от других частей проекта. Среда разработки IDEA поддерживает простой запуск тестов с удобной настройкой параметров. В текущей работе используется Junit версии 4.8.2.

Гибкость в первую очередь обеспечивается унифицированным созданием фикстур. Фикстуры помогают многократно использовать программный код за счет правила, которое гарантирует исполнение определенной логики до или после исполнения теста[3].

4.2.3 Библиотека журналирования Log4j

Разрабатываемый программный компонент интегрируется в систему управления охранной системой. К этой системе предъявляется требование ведение журналов, содержащих важную информацию о работе программы, необходимой

для диагностики и локализации неисправностей и ошибок. Для решения этой задачи используется библиотека Log4j версии 1.2.12. Её основным преимуществом является гибкость настроек, которые хранятся в специальном конфигурационном файле log4j.xml. При необходимости можно легко изменить логику ведения журналов не меняя исходный код проекта.

Журналирование заключается в вызове методов объекта класса Logger, отвечающих за нужный приоритет, и передаче в параметрах сообщения. В используемой библиотеке определены следующие приоритеты[4]:

- FATAL — произошла фатальная ошибка - у этого сообщения наивысший приоритет;
- ERROR — в программе произошла ошибка;
- WARN — предупреждение в программе что-то не так;
- INFO — просто полезная информация;
- DEBUG — детальная информация для отладки;
- TRACE — наиболее полная информация. трассировка выполнения программы. Наиболее низкий приоритет.

Логика ведения журналов описывается с помощью аппендеров. Аппендер — объект, который определяет: что нужно делать с журналируемыми сообщениями[4].

4.2.4 Система контроля версий

Список использованных источников

1) СТО 1.1.1.04.007.0814-2009. Единый унифицированный протокол обмена верхнего уровня [Текст].- Введ. 2009
2) Java [Электронный ресурс]. – Режим доступа : интернет : http://ru.wikipedia.org/wiki/Java. Дата обращения 21.05.2013
3) Гловер Э. Переходим на JUnit 4 [Электронный ресурс]. – Режим доступа : интернет : http://www.ibm.com/developerworks/ru/edu/j-junit4/section4.html. Дата обращения 21.05.2013
4) Принцип работы логеров и аппендеров [Электронный ресурс]. – Режим доступа : интернет : http://www.log4j.ru/articles/UmlExample.html. Дата обращения 21.05.2013

ПРИЛОЖЕНИЕ А

Исходнички

ПРИЛОЖЕНИЕ Б

Тестовые исходнички