Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего профессионального образования

**«Уральский федеральный университет
имени первого Президента России Б.Н. Ельцина»**

Институт естественных наук и математики

Департамент математики, механики и компьютерных наук

Разработка системы интеграции сетевых файловых хранилищ

Учебная практика студента 2 курса
Толстов А. Е.

Научный руководитель:
доцент
Волканин Леонид Сергеевич

Екатеринбург
2017

РЕФЕРАТ

Ключевые слова: OAUTH, WEBDAV, HTTP, СЕТЕВОЕ ХРАНИЛИЩЕ ДАННЫХ, СЕТЕВОЕ ФАЙЛОВОЕ ХРАНИЛИЩЕ.

Объект исследования – система интеграции файловых хранилищ.

Цель работы – разработка программы, выполняющей интеграцию сетевых файловых хранилищ с целью упрощения доступа.

В процессе работы проводилась разработка программного комплекса, направленного на снижение нагрузки на пользователя при работе с различными сервисами, предоставляющими возможность хранения данных.

Основные конструктивные и экономические показатели: высокая производительность при малых трудозатратах, удобство и безопасность при работе на ЭВМ обладающей неизвестным для пользователя статусом наличия вредоносного программного обеспечения, простота распределения файлов по конечным хранилищам данным.

Система может применяться для унифицированного доступа к различным файловым хранилищам из одного интерфейса.

ВВЕДЕНИЕ

Актуальность проблемы

На данный момент ЭВМ чрезвычайно плотно интегрированы в повседневную жизнь человека. Одной из важнейших задач, которые стоят перед любым пользователем ЭВМ – надежно и безопасно хранить данные. Разработанная система позволяет очевидным и удобным для пользователя способом выполнять операции по загрузке, скачиванию, администрированию файлов. Однако при этом система легко может быть изучена, переконфигурирована и расширена за счет гибкой системы плагинов и открытого исходного кода.

Существующие альтернативы

На данный момент существует несколько видов альтернативных решений. Некоторые из них приведены ниже:

1) Внешние системы хранения данных – чрезвычайно дороги и сложны в настройке и использовании. Неправильное развертывание может привести к долговременным проблемам с производительностью и потерей данных.
2) Платная подписка на сервисы хранения данных – требует постоянных денежных затрат для доступа к необходимому объему данных. Отсутствует гарантия защищенности данных от несанкционированного чтения, по причине невозможности добавить end-to-end шифрование.

Основные принципы

1) Открытость исходного кода программы
   Код программы распространяется под лицензией GNU GPL[1], а, следовательно, доступен для использования, изучения и модификации
2) Расширяемость программы за счет принципа SRP и паттерна DI
3) Надежность программы за счет разработки по паттерну TDD критически важных участков кода

Постановка задачи

Разработать приложение, позволяющее пользователю загружать и скачивать файлы, без необходимости выбора конечного сетевого диска: программа должна автоматически выбирать диск, на котором есть место для загрузки, либо диск, на котором есть файл при скачивании. При этом пользователь должен видеть содержимое диска, как будто он использует не несколько конечных сетевых дисков, а один. Все файлы, находящиеся в директории с одним и тем же названием, но на разных сетевых дисках должны отображаться пользователю как находящиеся в одной директории. При наличии конфликтов в

[1] Free Software Foundation, Inc. Стандартная общественная лицензия GNU (GPL). – 2007. − URL: https://www.gnu.org/licenses/gpl-3.0.ru.html (дата обращения: 25.05.2017).

конфигурации дисков, программа не должна пытаться их исправить, так как это может повлечь за собой потерю данных пользователем.

РЕАЛИЗАЦИЯ ЗАДАЧИ

Технические характеристики

Язык программирования – C#6
Программная платформа – .NET Framework 4.5.2
Требуемые операционные системы – Windows 7, 8, 8.1, 10
Минимальный требуемый объем RAM – 4Gb

Используемые приемы разработки

1) ООП – весь код написан в объектно-ориентированной парадигме
2) Разработка через тестирование (TDD[2]) – наиболее важные и ошибкоопасные места кода разрабатывались по данной технологии. Это позволяет избежать ошибок при дальнейшей разработке и экспериментально проверить корректность работы кода.
3) Принцип единственной ответственности (SRP) – все программные сущности, которые могут быть точками расширения имеют только одну причину для изменения и только одну обязанность.[3]
4) Средства автоматической сборки графа зависимостей – DI-контейнер – для подключения пользовательских плагинов используется средство рекурсивного сбора графа зависимостей, позволяющая прозрачно для прикладного разработчика подключить разработанный им плагин

Используемые технологии

1) Ядро и основное приложение:
    a. NUnit – средство для проведения модульного тестирования
    b. FakeItEasy – средство создания эталонных объектов-имитаций, реализующих абстракции – для тестирования кода независимо от реализации абстракций, которые он использует (Mock)
    c. log4net – система логирования
    d. WinForms – библиотека для разработки GUI для ОС Windows
    e. Ninject – средство сборки графа зависимостей
2) Стандартные плагины:
    a. HTTP – транспортный протокол.
    b. WebDAV [4]– набор расширений протокола HTTP, используемый для передачи файлов.
    c. YandexDiskSDK – базовое SDK, необходимое для доступа к Yandex диску. Было переписано, по причине неактуальности.

---

[2] Мартин Р. Чистый код: создание, анализ и рефакторинг – Санкт-Петербург, 2016. – С. 238.
[3] Там же. С. 30.
[4] Dusseault. L. HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV) // The Internet Engineering Task Force. – 2007. − URL: https://www.ietf.org/rfc/rfc4918.txt (дата обращения: 25.05.2017).

d. OAuth 2[5] - протокол авторизации пользователя, обеспечивающий защищенность его учетных данных (логина и пароля), в тоже время позволяющий выполнять разрешенные пользователем операции от его имени.

Описание структуры программного кода

Весь код делится на 3 группы (библиотеки/программы):

1) Код ядра (CloudMerger.Core) – данная библиотека содержит классы, которые могут быть использованы прикладным программистом для расширения существующего функционала.
2) Код тестов ядра (CloudMerger.Core.Test) – данная библиотека содержит код (Unit-тесты), позволяющие убедиться в корректной работе библиотеки ядра.
3) Код вспомогательных примитивов (CloudMerger.GuiPrimitives) – данная библиотека содержит вспомогательные для прикладного программиста классы, позволяющие ему формировать элементы GUI.
4) Код стандартных плагинов (hostings) – данная группа библиотек содержит базовые плагины и тесты к ним, позволяющие построить простейший хостинг без необходимости писать дополнительные плагины.
5) Код конечного приложения (ConsoleApplication) – данное приложение содержит код сборки графа зависимостей, а также простейшее консольное приложение, позволяющее пользователю выполнять минимальный набор операций – загрузку, скачивание, удаление файлов; создание и удаление директорий; получение информации об файлах и директориях.

Описание структуры скомпилированного приложения

1) ConsoleApplication.exe – основной исполняемый файл
2) *.dll – основные библиотеки
3) topology.ini – файл конфигурации учетных данных (Кодировка - ASCII)
4) hostings – директория, содержащая плагины в виде dll-библиотек

Конфигурирование скомпилированного приложения

В файле topology.ini содержится информация об учетных данный пользователя. Каждая строка файла представляет из себя:
[уровень вложенности][имя хостинга][идентификатор][OAuth-token]

1) Уровень вложенности показывает, в какое хостинг вложен данный хостинг. Один уровень вложенности отмечается одним символом «пробел» (код - 32)

[5] Hardt. D. The OAuth 2.0 Authorization Framework // The Internet Engineering Task Force. – 2012. − URL: https://tools.ietf.org/html/rfc6749 (дата обращения: 25.05.2017).

2) Имя хостинга – имя хостинга, советующее имени сервиса в плагине
3) Идентификатор – произвольная строка, позволяющая пользователю «опознать» данные учетные данные. Игнорируется при работе сервиса. Может быть заменена на символ «вопрос» (код - 63)
4) OAuth-token – токен, выданный пользователю для авторизации на определенном ресурсе. В случае хостинга, объединяющего другие хостинги – заменяется на символ «вопрос» (код - 63)

Подробное описание кода ядра (CloudMerger.Core)

1) HostingException – набор ошибок, которые могут быть сгенерированы хостингом в процессе работы
2) IHosting – интерфейс, опиывающий функционал хостинга: загрузку и скачивание файлов, а также базовые операции администрирования
3) IHostingManager – интерфейс описывающий конкретный сервис (например - YandexDisk). Предоставляет функционал создания реализации IHosting по учетным данным.
4) IMultiHostingManager – интерфейс описывающий хостинг, объединяющий несколько хостингов (например - MergedHosting). Предоставляет функционал создания реадизации IHosting по массиву вложенных IHosting-ов.
5) Пространство имен «Primitives» - описывает примитивы, использующиеся во всем приложении
    a. HostingTreeBuilder - примитивы построения дерева хостингов
    b. DiskSpaceInfo, ItemInfo, ItemType – примитивы, описвающие объекты файловой системы
    c. OAuthCredentials – примитив, хранящий учетные данные для определенного ресурса
    d. UPath – примитив, описывающий унифицированный путь в файловой системе
6) Tree – примитив, описывающий орграф дерево
7) Utility – вспомогательные классы и методы-расширения

Реализации интерфейсов IHosting и IHostingManager должны иметь конструктор без аргументов. Хотя бы один из этих интерфейсов должен реализовать прикладной программист, пишущий плагин к приложению. Написанный плагин и скомпилированный в dll-библиотеку плагин необходимо разместить в папке «hostings» конечного приложения вместе со всеми зависимостями (за исключением стандартных «Core» и «GuiPrimitives»)
Вся логика работы реализаций в случае неконсистентного состояния сетевых дисков приведена в XML-комментариях к IHosting и IHostingManager, описывающих выбрасываемые исключения и возможные на то причины.

Код программы

По причине большого объема, код не может быть приведен в данной работе в качестве приложения, однако он доступен по ссылке https://github.com/anaym/CloudMerger

ЗАКЛЮЧЕНИЕ

Разработанный комплекс в полной мере удовлетворяет поставленной задаче, код легко читаем (на основании метрик кода, приводимых средой разработки), а также хорошо документирован (за счет тестов и XML-документации)

СОДЕРЖАНИЕ

УСЛОВНЫЕ ОБОЗНАЧЕНИЯ

ЭВМ – Электронной вычислительная машина
.NET – .NET Framework
DI – Паттерн внедрения зависимостей
TDD – Паттерн разработки через тестирования
SRP – Принцип единственной ответственности
ООП – Объектно-ориентированное программирование
GNU GPL – Универсальная общественная лицензия GNU
RAM – Оперативная память
Gb – гибибайт. 1Gb = 1073741824 байт
GUI – графический интерфейс пользователя
ОС – Операционная система
SDK - Комплект средств разработки

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

Мартин Р. Чистый код: создание, анализ и рефакторинг. – Санкт-Петербург: Изд-во Питер, 2016.

Free Software Foundation, Inc. Стандартная общественная лицензия GNU (GPL) [Электронный ресурс] – 2007. − https://www.gnu.org/licenses/gpl-3.0.ru.html (дата обращения: 25.05.2017).

Dusseault. L. HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV) [Электронный ресурс] // The Internet Engineering Task Force. – 2007. – 2007. − https://www.ietf.org/rfc/rfc4918.txt (дата обращения: 25.05.2017).

Hardt. D. The OAuth 2.0 Authorization Framework) [Электронный ресурс] // The Internet Engineering Task Force – 20012. − https://tools.ietf.org/html/rfc6749 (дата обращения: 25.05.2017).