Агрегирование облачных хранилищ данных

Дмитрий Обыденков,
11 класс лицея информационных технологий №1537
Научный руководитель:
Минченко Михаил Михайлович,
учитель лицея информационных технологий №1537,
кандидат экономических наук

Использование облачных сервисов хранения данных становится сегодня объективной необходимостью, что обусловлено такими преимуществами этих сервисов, как обеспечение сохранности хранимых данных, отсутствие привязки к конкретному компьютеру, т. е. возможность доступа к хранимым данным через любой компьютер, подключенный к Интернету.

Причин, побуждающих пользоваться одновременно несколькими сервисами облачного хранения, несколько: отсутствие поддержки приложениями, использующими облачное хранение данных в различных целях (резервирование, обмен данными и т.д.); несовместимость ряда интернет-сервисов облачного хранения данных со сторонними облачными сервисами из-за противоречия политике продвижения собственных облачных сервисов; отсутствие приложений для работы с облачными сервисами для конкретных мобильных платформ из-за противоречий с политикой продвижения облачного ресурса или мобильной платформы или отсутствия интереса со стороны разработчиков к разработке приложений для мобильной платформы или ограничений мобильной платформы. Из-за перечисленных проблем пользователь не всегда может использовать единственный сервис облачного хранения данных и испытывает некоторые неудобства при пользовании несколькими сервисами.

Цель работы — разработка программного приложения, облегчающего взаимодействие пользователей с различными сервисами облачного хранения данных и повышающего надежность распределенного хранения данных.

Предмет работы — программная разработка, позволяющая одновременно использовать несколько облачных хранилищ: как отображение содержимого всех облачных хранилищ в виде «единого облачного пространства», так и отображение содержимого отдельного хранилища, а также возможность загружать данные, как используя одно облачное хранилище, так и используя «единое облачное пространство».

Основные задачи разработки:

- создание удобного интерфейса управления облачными хранилищами;
- автоматическое объединение содержимых облачных хранилищ с возможностью просмотра содержимого конкретного хранилища;

• просмотр содержимого конкретного облачного хранилища и «единого облачного пространства» с возможностью сортировки: по типу файла, по дате загрузки, по названию;

• возможность управления процессом распределенной загрузки в облачное хранилище: контроль степени распределенности, распределение по облачным хранилищам согласно командам пользователя;

• при необходимости пользователь должен иметь возможность отключить функцию распределенной загрузки данных и предоставить возможность выбора конечного расположения загруженного файла;

• предоставление пользователю возможности редактирования документа MS Office (документы форматов: *.doc, *.docx, *.ppx, *.xlsx, *.ppt) без необходимости скачивать файл на компьютер и, соответственно, без необходимости устанавливать пакет MS Office;

• предоставление пользователю возможности выбора директории расположения файла на компьютере пользователя при скачивании файла из облачного хранилища;

• автоматическая сборка распределенного файла в исходный при скачивании файла из «единого облачного пространства» без участия пользователя;

• возможность открыть файл непосредственно после завершения скачивания из облачного хранилища. Реализация в виде всплывающего сообщения после завершения загрузки «Скачать и открыть»;

• предоставление пользователю интерфейса управления используемыми облачными сервисами через форму «Менеджер аккаунтов»;

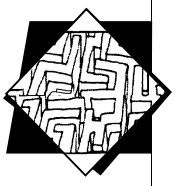
• возможность аварийного восстановления файлов в случае некорректной работы приложения;

• перспективные возможности сжатия и шифрования файлов;

• возможность предоставления пользователю выбора языка интерфейса приложения. По умолчанию используется русский.

Инструментальные средства и методы разработки

Основное инструментальное средство программной реализации — интегрированная среда разработки Microsoft Visual Studio 2012 и объектно-ориентированный язык C# с использованием компонентов .NET Framework. Для обмена данными с веб-хранилищами используются запросы Representational State Transfer (REST), возвращающие информацию в формате JavaScript Object Notation (JSON). Работа с данными в формате JSON выполняется при помощи открытого фреймворка JSON.NET. Для реализа-



ции пользовательского интерфейса использована система Windows Presentation Foundation на основе XAML.

Основным преимуществом выбора связки класса WebClient, HttpRequest и запросов REST перед пакетом Live Software Development Kit являлась возможность поддержки стандартных методов HTTP — таких, как GET, PUT, POST и DELETE, т. е. возможность унификации методов обращения к различным сервисам облачного хранения данных. Безопасность обмена данными обеспечивает также общий протокол OAuth — открытый стандарт проверки подлинности учетных данных пользователей.

Информация о каталоге облачного хранилища передается в формате Json. Средства фреймворка JSON.NET позволили использовать SQL-подобный язык Language Integrated Query (LINQ) для составления запросов к структуре облачного хранилища. Использованы также методы фреймворка для сериализации, десериализации и, конвертации XML в JSON полученных данных.

Для выполнения программной реализации изучены:

- особенности объектно-ориентированного языка программирования C# и языка интегрированных запросов LINQ;
- асинхронное программирование;
- разметка интерфейса приложения при помощи языка XAML;
- особенности работы с форматом данных JSON;
- особенности сетевых взаимодействий: авторизация и обмен данными между сервером и клиентом;
- особенности многопоточных приложений.

Одной из ключевых особенностей программы является ***возможность создания «единого облачного пространства»*** (далее ЕОП). За основу взят принцип организации дисковых массивов RAID 0, но, в отличие от дискового массива, где в качестве носителей информации используются физические накопители (например: HDD, SSD), ЕОП в качестве носителя информации использует удаленный сервер, предоставляемый сторонним сервисом. Как известно, самым большим недостатком RAID 0 является низкая отказоустойчивость — в случае отказа одного накопителя в массиве нерабочей становится вся система. ЕОП лишена этого недостатка, поскольку безопасность информации гарантируется сервисом облачного хранения данных.

Особенностью RAID 0 является более высокая производительность — ЕОП так же имеет большую скорость доступа к данным. Решение позволяет добиться скорости загрузки, равной сумме скоростей загрузок на каждый сервер, и снизить время ожидания к минимальному значению. При использовании большого количества различных серверов возможно достижение максимальной скорости загрузки, ограниченной исключительно про-

вайдером. По умолчанию, файл делится на равные по размеру части, но после сбора статистики о скорости соединения с каждым сервером появляется возможность делить файл на неравные части. Соответственно, распределение частей файла по облачным хранилищам в зависимости от размера части и статистической скорости соединения позволит дополнительно сэкономить время загрузки. Еще одной возможностью, позволяющей снизить время ожидания, является предварительное сжатие данных перед распределенной отправкой данных в облачные хранилища. Данная функция будет работать эффективно только для данных, поддающихся сжатию — таких, как текстовые документы и растровая графика.

В основе алгоритма реализации «единого облачного пространства» лежит следующий принцип: на запрос о содержимом облачное хранилище возвращает перечень элементов с сопутствующей информацией; в результате обработки списков программа предоставляет пользователю совокупность в виде «единого облачного пространства». При нахождении совпадений в файлах и папках, то есть при нахождении распределенных файлов и папок, соответствующие элементы отображаются, как обычный файл. Для всех распределенных объектов при отображении в списке устанавливается отметка, позволяющая отличить их от нераспределенных. Таким образом, приложение встраивается в облачное хранилище незаметно для пользователя.

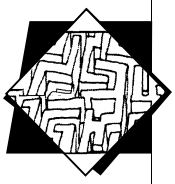
Программная реализация пользовательского интерфейса

Пользовательский графический интерфейс (GUI) реализован на основе библиотек визуальных компонентов и функциональных возможностей Win32 API. ***Структура пользовательского интерфейса*** включает в себя следующие экранные формы (приложение, рис. 1–4):

1. ***Главная форма*** появляется на экране при запуске приложения и выполняет функцию структурированного вывода данных, работы с ними и общего управления приложением.

Для работы с данными используются элементы:

• Для вывода списка файлов, содержащихся в выбранном каталоге, используется элемент ListView в режиме таблицы. Таблица содержит столбцы «Имя файла», «Размер», «Расположение файла» (отображает название облачного хранилища, в котором содержится файл; в случае, если файл распределен, то отображаются хранилища, по которым распределен файл.), «Дата загрузки» в которых отображаются соответствующие свойства файла. При нажатии на заголовок столбца данные сортируются по возрастанию, а при повторном клике — в обратном порядке. Данный



элемент занимает большую часть пространства на форме, так как является основным средством работы с данными.

• Для вывода списка папок, содержащихся в выбранном каталоге, используется элемент TreeView. Данный элемент интерфейса отображает каталоги в зависимости от выбранного облачного хранилища, при выборе режима отображения распределенного хранения отображаются совокупность всех подключенных облачных хранилищ.

• В правом углу содержится информация (название, размер, количество содержащихся элементов) о выбранном каталоге и информация выбранном файле.

• Полоса меню содержит несколько ключевых пунктов:

— «Загрузка файла» — открывает форму «Опции загрузки файла».

— «Быстрая загрузка файла» — вызывает OpenFileDialog для быстрой загрузки файла в соответствии с заранее заданными параметрами. Если параметры ранее не были заданы, будет вызвана форма «Настроить быструю загрузку файлов», аналогичную по функционалу форме «Опции загрузки файла».

— «Скачать файл» — скачивает файл из облачного хранилища в каталог, указанный SaveFileDialog. При завершении скачивания файла всплывает диалоговое окно, предоставляющее пользователю выбор: «Открыть файл» или «Отменить».

— Для нераспределенных файлов (несжатых и незашифрованных), хранящихся в облачных хранилищах, поддерживающих веб-версию MS Office — предоставление быстрого редактирования файла в браузере. Нажатие на «Открыть файл в веб» открывает браузер с веб-версией редактора.

— «Менеджер аккаунтов» — вызывает форму «Менеджер аккаунтов».

— «Аварийное восстановление файлов» — вызывает форму «Аварийное восстановление файлов» в случае, если автоматическая сборка файла по каким-то причинам не работает, позволяет вручную собрать файл.

— «Язык» — позволяет выбрать язык интерфейса (русскоязычный/англоязычный).

2. ***Форма «Менеджер аккаунтов»*** (приложение, рис. 3) — позволяет подключить новое облачное хранилище или удалить из приложения уже добавленное.

• Основным элементом является *элемент ListView*, который отображает список подключенных облачных хранилищ в виде списка элементов, каждый из которых включает в себя логотип облачного хранилища, его название, присвоенное ему пользователем, и элемент интерфейса, наглядно отображающий степень наполненности. Двойной щелчок по элементу вызывает вспомогательную форму, позволяющую изменить данные облачного хранилища.

• *Кнопка «Добавить»*, содержащаяся на форме, вызывает браузер, позволяющий подключить новое облачное хранилище. В целях безопасности приложение не имеет доступа к паролям пользователя, при авторизации пользователь разрешает приложению использовать облачное хранилище.

• *Кнопка «Удалить»* исключает облачное хранилище из списка используемых.

3. ***Форма «Опции загрузки файла»*** устанавливает параметры загрузки файла в облачное хранилище. В отличие от пункта меню «Быстрая загрузка файла», при котором нет необходимости каждый раз применять параметры, так как используются заранее заданные параметры, позволяет в случае необходимости использовать параметры, отличные от базовых.

• Главный параметр, который предположительно пользователь будет чаще всего изменять: распределенное или нераспределенное хранение (также выбрать хранилища, по которым будет распределен файл). Данный функционал реализован через элемент CheckList, в котором пользователю нужно будет поставить галочки напротив хранилищ, в которых будет содержаться файл. Соответственно, при нераспределенном хранении пользователю надо будет поставить единственную галочку напротив хранилища (из списка ранее добавленных), в которое загружается файл, а при распределенном хранении — указать группу облачных хранилищ.

• Кнопка «Добавить» вызовет форму OpenFileDialog.

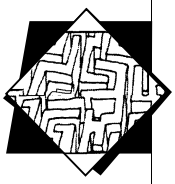
• Элемент ChechBox: «Сжатие». При наличии соответствующей отметки файл перед загрузкой в облако будет сжат.

4 Для случая непредвиденной ситуации или некорректной работы приложения, в результате которой после распределенного хранения пользователь может не получить исходный файл, создан ***модуль «Аварийное восстановление файлов».*** Пользователю необходимо указать каталог через вызов формы OpenFolderDialog, в котором содержатся части файла в формате <название файла>.<расширение>.part????, после чего модуль, в случае успешной сборки файла, будет помещен в указанную директорию.

При успешном выполнении операции или возникновении ошибки будет отображено соответствующее сообщение.

Выводы и преимущества разработки

В результате разработано приложение, обеспечивающее формирование «единого облачного пространства», составными частями которого являются зарезервированные накопительные пространства, принадлежащие различным интернет-сервисам. Пользователю предоставляется одновременный доступ ко всем данным, содержащимся в пространствах, принадлежащих различным интернет-сервисам. Разработанное приложение позволит со-



кратить время поиска необходимого файла, поскольку оно предоставляет интерфейс для работы одновременно со всеми источниками. Основные преимущества облачных хранилищ как накопителей — надежность, безопасность, универсальность — позволяют назвать приложение решением для повышения надежности хранения данных.

Такие особенности разработанного приложения, как возможность одновременной работы с несколькими хранилищами и функция распределенной загрузки, позволяют выделить его среди существующих аналогов. Кроме того, ***основными преимуществами*** разработанного приложения являются:

- создание общего накопительного пространства;
- снижение времени загрузки посредством распределения нагрузки на несколько каналов;
- снижение минимального времени, необходимого для администрирования;
- возможность бесшовного наращивания объема общего дискового пространства;
- адаптация облачных технологий к более широкому слою пользователей;
- обеспечение сохранности данных (надежность);
- отсутствие привязки к определенному накопителю.

Системные требования для оптимальной работы приложения

Разработанное приложение ориентировано на использование на ПК с операционной системой Windows XP и выше, обязательно установленным .NET Framework 4.5. Для полноценной работы с приложением также необходимо наличие аккаунтов в сервисах облачного хранения данных.

Приложение

Форма авторизации и подтверждения права доступа приложения к облачному хранилищу SkyDrive

Рис. 1. Форма авторизации и подтверждения права доступа приложения к облачному хранилищу Dropbox

Рис. 2. Окно «Менеджер аккаунтов» — для добавления новых облачных хранилищ в приложение

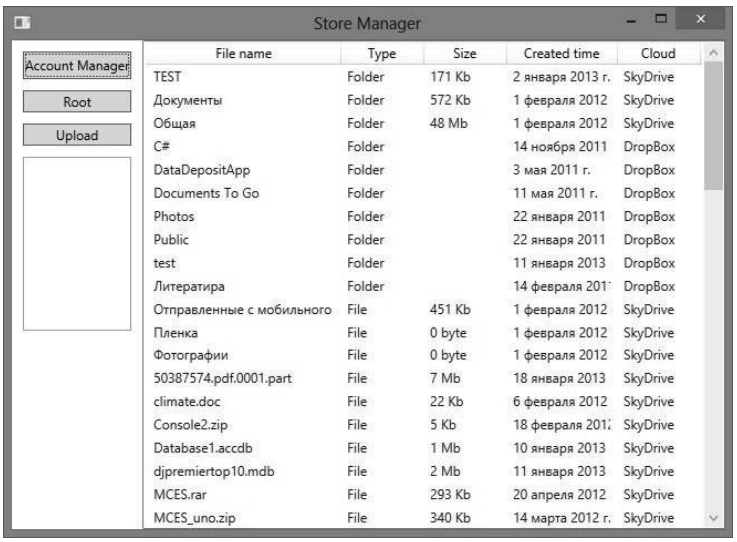


Рис. 3. Основное окно — для отображения содержимого облачных хранилищ

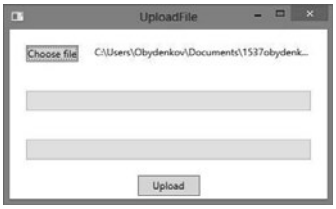


Рис. 4. Окно загрузки файла: загрузка происходит одновременно в два облачных хранилища — поэтому имеется два индикатора прогресса загрузки