Файловые системы и базы данных

Слободан Целенкович (Slobodan Celenkovic), Вторник, 15 Март 2005, 22:04

Основная идея

Хотя и существует множество вариаций, суть идеи в том, чтобы расширить количество и типы атрибутов файлов, чтобы позволить находить и получать доступ к файлам без знания точного пути. BeOS добавляла B-tree индексы для дополнительных атрибутов (таких как автор, название, и т.д.), тем самым делая возможным быстрый поиск, основанный на этих атрибутах. Похоже, что WinFS будет использовать для хранения и индексации атрибутов файлов Microsoft SQL server (или его версию).

Использование технологии реляционных СУБД для индексации атрибутов файлов позволяет проводить быстрый поиск файлов, основанный на нескольких атрибутах, отличных от имени файла и пути к нему. В результате, путем добавления новых способов доступа, которые не были доступны ранее, мы повышаем доступность данных. По сути, идея добавления базы данных к файловой системе направлена на добавление множества новых методов доступа к файлам. Повышение гибкости позволяет пользователям находить файлы, используя различные описания, в особенности, описания специфичиные для данной конкретной области, которые не имеют ничего общего с именем файла и его пути.

Аргументы против

Многие разработчики и другие технически подкованные пользователи компьютеров серьезно возражают против этой идеи. Однако, зачастую их мнения не подкреплены достаточной информацией и не обладают ясностью. Легко делать выводы, основываясь только на заголовках или кратких описаниях, приводимых в новостях.

Основной огонь критики направлен на то, что добавление базы данных просто не является необходимостью. В конце концов, все современные файловые системы уже работают очень хорошо, обеспечивая очень быстрый и эффективный доступ ко множеству файлов. Зачем возиться с ними? Это философия "не чини не ". Это абсолютно правильный аргумент, так как хорошо известно, что сегодняшние файловые системы - результат долгой эволюции, и поэтому они обладают многими прекрасными характеристикам. Тем не менее, важно отметить то, что идея просто добавляет базу данных к существующей файловой системе, а не отменяет ее! Стандартная файловая система продолжает играть ту же роль, что она играла всегда. Ее не удаляют и не заменяют!

Еще один аргумент состоит в том, что накладные расходы, которые принесет база данных поверх (или совместно) файловой системы просто не нужны. У этого аргумента есть несколько слабостей. Во-первых, сегодня накладные расходы еще не до конца ясны. Этот аргумент может стать актуальным только если накладные расходы будут значительны. Реально же, все реализации стараются минимизировать накладные расходы. В прошлом, когда вычислительные ресурсы были дефицитом, этот аргумент имел смысл. Доступная сегодня массовая обработка данных, емкости памяти и жестких дисков, должны справиться с накладными расходами без каких-либо проблем. В частности, будущие многоядерные процессоры смогут легко запускать операции с базой данных на одном из ядер в фоновом

режиме с небольшим или совсем без ущерба другим задачам.

Еще один аргумент заключается в том, что база данных добавляет к файловой системе просто ненужную сложность. Существует множество реляционных СУБД разных размеров и сложностей. В общем случае, добавляемые к файловым системам СУБД выбираются из тех, что поменьше и попроще. Мы не говорим о мощной СУБД Oracle, использующей огромные объемы памяти и дискового пространства. В основном это будут какие-либо небольшие встраиваемые системы управления базами данных, которые значительно менее требовательны к ресурсам машины.
[newpage]

Перспективы

Принципиальный аргумент против этого - то, что нам оно просто не нужно. Уточню, нам - читателям, программистам, техническим пользователям - не надо. Конечно, есть и другие пользователи, другие взгляды, которые могут не быть хорошо представленнымиздесь. В частности, менее умудренным пользователям компьютеров, на самом деле, это необходимо.

Все большее число людей использует компьютеры. Они одновременно производят и используют все возрастающее количество файлов. Сети также послужили причиной того, что эти файлы разбросаны по нескольким компьютерам, от файловых серверов компаний до Palm Pilot'ов и iPod'ов. Пользователей мало интересует эффективность алгоритмов, накладные расходы, создаваемые базами данных, и другие технические аспекты. Они заинтересованы в том, чтобы их время не тратилось на поиски файлов. Проблема упорядочивания и нахождения файлов будет становиться только больше, никак не меньше!

Эти пользователи не работают в технических областях и не знают концепции путей и имен файлов. Каждый раз, когда им приходится использовать файловый менеджер или диалоговое окно открытия файла, им необходимо переключиться на другую область. Хуже того, большинство из них не имеет ни желания, ни способности нормально упорядочить файлы. Как часто вы видели компьютер новичка с рабочим столом, увешанным иконками файлов? Зачастую они даже не знают или не понимают каталогов. Мой отец был в панике, когда файл вышел из списка недавно использованных в Word. Он просто никогда не использовал файловый менеджер.

Полные пути файлов - просто уникальные ключи файлов, необходимые для того, чтобы обращаться к файлам без каких-либо неоднозначносте. Они играют ту же роль, что и первичные ключи в системах управления базами данных. Есть и другие системные атрибуты (несколько дат, списки контроля доступа (ACL - Access Control List), но никаких других атрибутов из конкретной области. Базы данных могут хранить и управляться со множеством других атрибутов, так что добавлять множество атрибутов, касающихся заданной области, просто. Помимо стандартного автора, заголовка и подобного, в офисе доктора могут добавить атрибуты, относящиеся к здоровью, в офисе адвоката - добавить атрибуты, относящиеся к законам, и так далее.

На самом деле, нет причин, по которым один тип атрибутов должет быть доминирующим. Имя файла должно быть всего лишь одним из множества равноправных атрибутов. Тогда графический интерфейс, такой как файловый менеджер, должен позволять пользователям осуществлять навигацию, используя другие атрибуты. Например, он может показывать иерархию, где на первом уровне идут авторы, а на втором - заголовки. В реальности,

возможно множество различных комбинаций, но это отдельная статья.

Возможность определения дополнительных атрибутов из области и управление ими (хранение, поиск,...) на уровне ОС снимает груз с приложений. К тому же она дает возможность создания стандартизированых, независимых от приложений, механизмов доступа к атрибутам файлов. Сравните это с сегодняшней ситуацией, когда поиск информации по множеству типов файлов и бинарных форматов, очень сложен и мучителен.

Итог

Программисты склонны и имеют возможность разрабатывать всевозможные инструменты для упорядочивания своих файлов. Мы просто решаем свои проблемы путем создания новых инструментов. Более обширное сообщество пользователей не может сделать этого. Они нуждаются в помощи. С их точки зрения проблема управления большим количеством файлов существует уже сегодня. Поэтому нам пора двигаться от вопросов о том, необходимо ли это, к выбору лучших архитектур/реализаций решения.

Смотрите также: В качестве примера файлового менеджера, основанного на этих и других идеях посмотрите на Dekk.