УДК 004.75

ПРОГРАММНЫЕ ПОДСИСТЕМЫ КЛАСТЕРА НАДЕЖНЫХ ВЫСОКОСКОРОСТНЫХ ХРАНИЛИЩ ДАННЫХ БОЛЬШИХ ОБЪЕМОВ

А.Л. Надточей
(представил д.т.н., проф. В.С. Харченко)

Рассмотрены структура и взаимодействие программных подсистем кластера, предназначенного для хранения данных больших объемов. Определены параметры кластера, которые обеспечивают надежную и высокоскоростную работу кластера.

Постановка проблемы. Главной задачей современных информационных систем является обработка информации, представленной в различном виде. Часть этой информации структурируется и представляется в виде таблиц баз данных, которые с каждым годом хранят все большее количество данных. Интерфейс этих данных с пользователем является одним из главнейших звеньев информационной системы. Однако, возможности любой информационной системы, построенной таким образом, ограничены. Ценность информации, которая с каждым днем возрастает, не позволяет свободно расширять функциональность или же заменять систему на новую.

Наряду со строго структурированной информацией, в любой системе присутствует огромное количество дополнительных данных, хранимых в виде файлов. Любая информационная система строится по строгим и упорядоченным принципам, где каждый пользователь системы работает в рамках отведенных ему ресурсов и его действия ограничены рамками предоставленных ему полномочий. Современные операционные системы предоставляют все необходимое для реализации различных стратегий построения информационных систем. Одним из самых важных правил является *централизованность обработки и хранения данных*. Реализация именно этого правила позволила системам управления базами данных (СУБД) занять центральное и лидирующее положение в информационных системах. Для СУБД разработан и практически реализован целый ряд специализированного программного обеспечения (СПО), которое осуществляет резервное хранение данных. В алгоритмы функционирования СУБД внесены части, отвечающие за полное или частичное восстановление информации в случае какого-либо отказа или сбоя [1].

Хуже обстоят дела с файловыми системами. Централизованное хранение файлов реализуется с помощью файловых серверов, где алгоритмы файловых систем направлены на максимально быструю обработку запросов пользователей по чтению или записи файлов. ***Надежность хранения информации*** – фактор важный, но стоит вторым в очереди после производительности. Поэтому необходимо использовать дополнительные средства, направленные на уменьшение вероятности потери информации, которая хранится в файлах. Одним из таких средств является *подключение файловых серверов через источники бесперебойного питания*. Если такой метод позволяет уменьшить вероятность потери информации при сбоях в питании компьютеров, то он не страхует пользователя от потери информации при отказе сервера из-за механических повреждений. Вторым стандартным методом является *резервное копирование*. К сожалению, не во всех современных операционных системах реализовано программное обеспечение, которое позволяет решить проблемы резервного копирования одним из стандартных методов. Второй трудностью, с которой рано или поздно сталкиваются пользователи информационных систем – это ***ограниченность дискового пространства***. Если для обычных сотрудников это чаще всего неудобство, которое связано с необходимостью своевременной чистки отведенного им пространства, то для администраторов – это вечная «головная боль». Очень важным фактором в любой информационной системе является ее ***внешнее постоянство для пользователя***. Другими словами, если программное обеспечение сотрудника предприятия настроено на работу с определенной базой данных, то какие бы манипуляции не производил обслуживающий персонал, это не должно влиять на работоспособность программы сотрудника. Учитывая, что такое понятие как подключение к базе данных, является абсолютно логическим и не зависит от аппаратуры компьютера, то реализовать его постоянство несложно. Если в большинстве современных СУБД проблемы добавления дополнительных мегабайт решены, то абсолютно по-другому выглядит ситуация с расширением дискового пространства, предназначенного для хранения файлов пользователей.

Таким образом, при управлении дисковыми массивами возникают следующие трудности:

– обеспечение надежности хранения информации;

– обеспечение минимального времени обработки запросов пользователей по чтению или записи;

– прозрачная с точки зрения работы пользователя структура, которую можно расширять при необходимости.

Анализ литературы. Рассмотрим средства, которые доступны администраторам компьютерных сетей по управлению дисковыми массивами данных. За основу возьмем операционную систему (ОС) Microsoft (MS) Windows, которая является самой популярной клиентской средой в настоящее время. В качестве стандартного решения, которое используется на предприятиях, является создание сетевых дисков и их последующее подключение к рабочим станциям пользователей [2, 3]. В течение долгого времени пользователь привыкает к своему рабочему окружению и любое его изменение приводит к неудобствам. Поэтому решение расширения дискового пространства за счет добавления нового накопителя на жестких дисках и последующее его предоставление в сеть в качестве дополнительного сетевого диска [3], на котором пользователи могли бы хранить информацию, чаще всего является самым простым и самым ошибочным. Пока данные пользователя хранятся через единый дескриптор доступа (будь-то имя каталога или же имя сетевого диска), он вполне нормально работает и не испытывает особых трудностей. Введение дополнительного диска переносит на пользователя задачу принятия решения о месте хранения информации, что впоследствии приводит к трудностям поиска сохраненного на одном из дисков файла. Попытки оставить среду пользователя прежней за счет применения дополнительных средств, которые бы работали прозрачно для пользователя, упираются в ограниченность функций операционных систем. В ОС MS Windows 2000 реализована поддержка распределенной файловой системы (DFS) [2, 3], которая является аналогом более старой сетевой файловой системы (NFS) операционных систем Unix. DFS позволяет представить диск в виде каталога. Для его поддержки на компьютерах пользователей необходимо установить дополнительное программное обеспечение и поменять дескриптор доступа с сетевого диска на каталог, что приводит ко временным трудностям в работе пользователя. Но и введение этой системы не решает до конца проблем надежного хранения файлов и свободной масштабируемости системы как в сторону увеличения, так и в сторону уменьшения объемов дискового пространства файловых серверов.

Цель статьи. В [4] были описаны основные концепции построения системы хранения файлов, в которой за основу была взята кластеризация. Была сделана попытка показать все позитивные стороны такой системы и рассмотрена ее архитектура. Целью же данной статьи является ***обоснование состава программных средств кластера надежного хранения информации больших объемов***.

Подсистемы кластера. В [4] было определено, что для повышения надежности функционирования кластера он должен быть децентрализо-

ванным. Следовательно, на каждом компьютере, входящем в кластер, должно быть установлено программное обеспечение, готовое в любой момент времени, вне зависимости от состояния кластера, принять решение о сохранении файла. Назовем это программное обеспечение ***диспетчером*** и отведем ему верхнюю ступень иерархии. Именно диспетчер будет принимать решение о месте сохранения поступившего файла, в соответствии с настройками кластера и его текущим состоянием.

После принятия решения о месте расположения файла над ним выполняется операция сохранения. Время выполнения этой операции зависит от размера файла и загруженности компьютерной сети кластера операциями сохранения других файлов, поступивших на другие узлы кластера. Поэтому диспетчер не может выполнять операцию сохранения файла самостоятельно. Следовательно, для выполнения операции сохранения файла поток байт должен быть передан дополнительной подсистеме, которая выполнит операцию сохранения и затем сообщит об ее окончании диспетчеру, который, в свою очередь, сообщит об этом компьютеру пользователя. Согласно алгоритмам функционирования протокола TCP/IP возможна одновременная передача информации по одному каналу связи нескольким программам. Это значит, что на узел кластера могут поступать потоки байт, которые принадлежат различным файлам. Выделение операции сохранения файла в отдельную подсистему решает эту задачу. Физически подсистема сохранения файла может быть представлена как отдельный процесс, который запускается диспетчером при получении запроса о сохранении файла, или же который запускается диспетчером после старта. Второй подход позволяет повысить производительность системы за счет отсутствия операций чтения и инициализации процесса, но требует дополнительных ресурсов. Также существует проблема указания числа процессов подсистемы сохранения, которые должны быть загружены в память. Выходом из сложившейся ситуации является использование смешанного метода, при котором при старте диспетчер загружается в память и инициирует работу нескольких процессов сохранения информации. По мере поступления запросов пользователей и отсутствия свободных процессов сохранения информации диспетчер может загрузить в оперативную память дополнительные процессы сохранения информации, передать им поток байт и после окончания операции сохранения, удалить дополнительный процесс из памяти. Однако, чтобы производительность кластера была оптимальной и не понижалась за счет большого числа процессов сохранения, следует внести параметр «*максимально допустимое число процессов сохранения*». На основании статистики работы кластера его администраторы могут корректировать это число. Чтобы исключить многократные операции инициализации процесса, которые возможны

в случае увеличения среднего числа запросов сохранения за единицу времени, следует внести параметр «*время бездействия процесса сохранения*», который определяет максимальное время нахождения в оперативной памяти дополнительного процесса сохранения при отсутствии запросов на сохранение файлов. Этот параметр позволит системе адаптироваться к текущему среднему количеству запросов.

Как было показано в [4], надежность кластера определяется избыточным копированием информации не только на компьютер-приемник, но и на другие компьютеры кластера. При этом файлы, которые передаются для резервного копирования, сжимаются одним из упаковщиков с целью увеличения дискового пространства. Число копий файлов определяет степень восстановления информации и возможность сохранения работоспособности, пусть и с худшим качеством кластера, при отказе какой-то его части. Следовательно, процедура сохранения файлов состоит из двух частей:

– сохранение файла на жестком диске компьютера – приемника;
– прием и передача упаковщику потока байт из сети.

Подготовка выполнения операции сохранения файла на других узлах кластера выполняется диспетчером с учетом загруженности узлов кластера. Количество резервных копий файла определяет один из важнейших параметров всего кластера, который называется «степенью резервного копирования». В соответствии с этим параметром диспетчер ищет число наиболее свободных узлов кластера. Таким образом для подготовки операции записи информации диспетчер должен выдать запрос всем узлам кластера, по которому те должны вернуть свой статус. С целью уменьшения числа передаваемой информации и упрощения алгоритмов принятия решения такой запрос выдается с использованием технологии широковещательных запросов сети и воспринимается всеми узлами кластера одновременно. Затем каждый диспетчер рассчитывает числовой коэффициент, характеризующий его загрузку, и передает его узлу, выдавшему запрос. Этот узел выполняет сортировку принятых коэффициентов и выбирает необходимое число компьютеров, реализующих параметр «*степень резервного копирования*» (кратность резервирования). Затем он отправляет выбранным компьютерам подтверждение на выполнение операции принятия потока байт, а всем остальным сообщает об освобождении их от операции сохранения файла. Процесс сохранения файла передает поток байт в сеть одновременно с его записью на диск. Таким образом, диспетчер, получивший запрос на сохранение файла, становится ведущим в кластере до окончания операции сохранения и, в свою очередь, в этот же момент может быть ведомым для операции сохранения другого файла другого узла кластера. Такой механизм позволяет реа-

лизовать децентрализованность кластера. Число процессов сохранения копии файла определяется настройками компьютера кластера, который инициирует запись файла в кластер и, следовательно, не требуется дополнительного параметра, определяющего количество процессов сохранения резервных копий. Поскольку процесс архивирования требует определенных вычислительных затрат, то параметр «*коэффициент загрузки узла кластера*», определяет не только возможность записи данных этим узлом, но и равномерность загрузки компьютеров кластера.

Для обработки запрошенной операции чтения диспетчеру требуется определить местонахождение файла. Если на диске данного компьютера файла нет, то диспетчер с помощью запроса определяет местонахождение файла, связывает свой процесс чтения файла с процессом чтения файла локального компьютера и предоставляет файл пользователю. При этом диспетчер увеличивает счетчик числа опросов файла с целью проведения последующей балансировки кластера.

Операции определения отказа узла кластера также выполняются диспетчером. С каждым посланным широковещательным запросом ассоциируется счетчик времени, по истечении которого при отсутствии ответа узел кластера считается недоступным. Для реализации описанного в [4] свойства кратковременного отключения узла кластера с целью проведения регламентных работ необходимо ввести параметр «*время ликвидности узла кластера*», который определяет интервал времени, в течение которого неотвечающий узел кластера считается рабочим. В течение этого времени файлы узла предоставляются с резервных копий, которые хранятся на других доступных узлах. По истечении этого времени узел считается неисправным и выполняется локальная балансировка кластера с целью распаковки и сохранения распакованных файлов кластера на других узлах согласно первым пришедшим запросам пользователей. Если узел кластера заработал спустя некоторое время после принятия решения об его неисправности, то необходимо выполнить синхронизацию данных узла с информацией кластера, используя дату последней модификации файлов.

Балансировка кластера выполняется диспетчерами и подразумевает перераспределение файлов по узлам кластера с учетом частоты обращений к файлам. Поскольку балансировка связана с большими затратами процессорного времени на распаковку файлов и их перемещение по сети, то *режим полной балансировки* является специальным и должен выполняться в часы наименьшей загрузки. Выполнение балансировки не должно приводить к временной остановке операций чтения или записи файлов, но предполагает значительное снижение производительности кластера в момент операции. Наряду с полной балансировкой может

быть выполнена *локальная балансировка* отдельного файла или файлов в случаях выхода из строя узла кластера или же превышения числа обращений к файлу какого-то порогового значения, что приводит к вынужденному перемещению файла на тот узел кластера, откуда к нему сделано наибольшее число обращений. Каждая операция балансировки сбрасывает счетчик числа обращений к файлу. Следовательно, во время интенсивной работы кластера реализуется некоторая адаптация кластера к задачам пользователя, которая существенно не влияет на производительность кластера в целом.

Выводы. ***ПО кластера*** должно состоять из следующих частей (рис. 1):

– ***диспетчера***, выполняющего функции принятия решения о сохранении файла или его чтении и использующего для выполнения операции соответствующий процесс;

– ***процесса сохранения***, работающего в двух режимах: сохранения файла и сохранения резервной копии файла;

– ***процесса чтения***, работающего в двух режимах: чтения файла и чтения файла из резервной копии.

Для нормального функционирования кластера его администратор должен задать следующие параметры:

– *степень резервного копирования*, которая определяет количество резервных копий и устойчивость кластера к отказам какой-то из его частей;

– *стартовое количество процессов сохранения*, которое предназначено для уменьшения времени получения максимальной производительности узла за счет принудительной загрузки в оперативную память указанного числа процессов сохранения;

– *максимальное количество процессов сохранения*, которое ограничивает максимальное число процессов сохранения;

– *стартовое количество процессов чтения*, которое, как и второй параметр, влияет на уменьшение времени готовности кластера к работе с максимальной производительностью;

– *максимальное количество процессов чтения*;

– *время ликвидности узла кластера* (время, по истечении которого недоступный узел кластера считается нерабочим);

– *порог локальной балансировки* – задает число обращений к файлу, при достижении которого он перемещается на компьютер, ближайший к клиенту, запрашивающему этот файл (локальная балансировка).

На момент выполнения операции записи или чтения диспетчер данного узла становится ведущим для данной операции. При этом он может быть ведомым для других операций записи или чтения, которые проводят другие узлы кластера и ведущим для других операции записи или чтения, которые

не завершились в данный момент времени. Это свойство позволяет реализовать децентрализованность кластера и повышает его надежность.

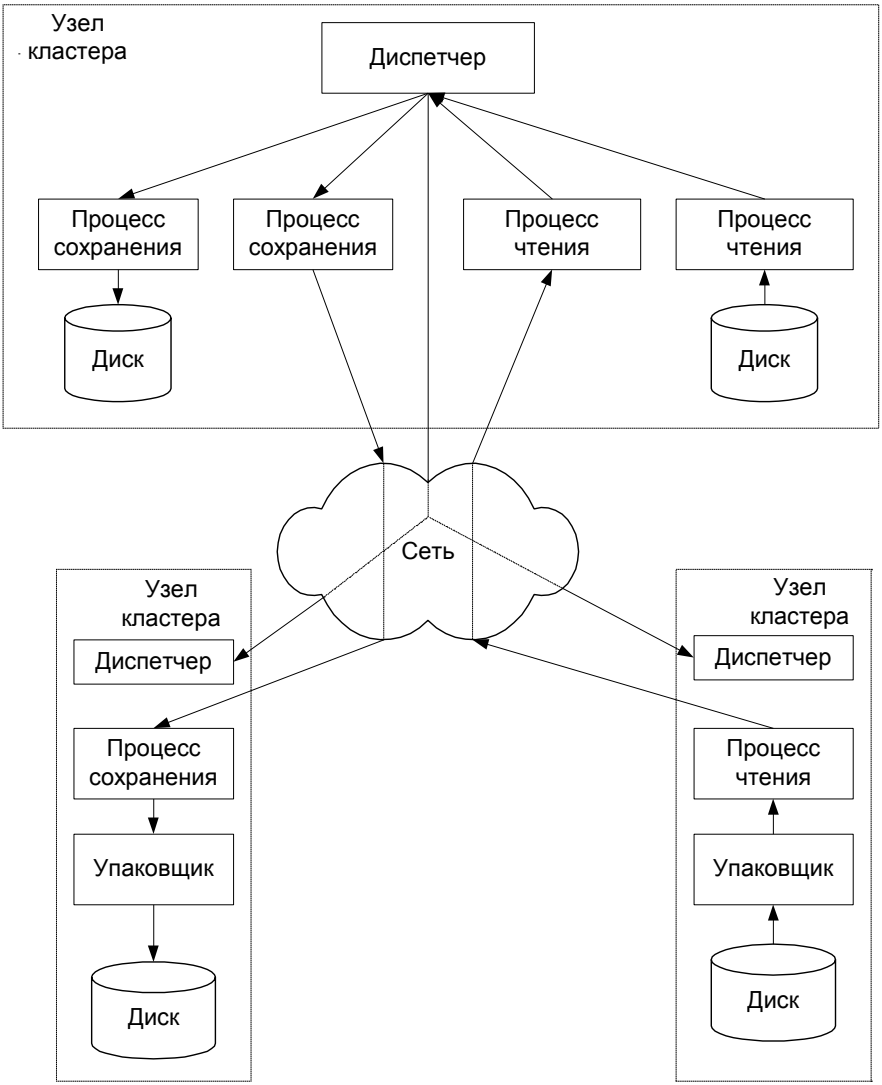

Рис. 1. Структура программного обеспечения кластера

.

Числовой коэффициент, с помощью которого принимается решение об использовании данного узла в качестве хранилища резервной копии

файла, должен учитывать загрузку процессора, оставшееся место на диске, число операций записи или чтения, которые выполняются в данный момент времени. Вычисляется коэффициент узлом кластера по поступлению запроса о сохранении файла и передается ведущему компьютеру. Алгоритм вычисления коэффициента должен быть простым и не требовать от узла кластера больших вычислительных затрат. Но с другой стороны, он должен давать четкое количественное значение загрузки узла.

Для каждого файла ведется счетчик числа обращений, который сбрасывается через определенные промежутки времени. При достижении счетчиком определенного порогового значения может быть принято решение о локальной балансировке, которая заключается в перемещении файла на узел кластера, с которого поступает наибольшее количество запросов.

Наряду с локальной балансировкой существует операция полной балансировки кластера, которая является ресурсоемкой и может приводить к замедлению работы кластера, но не останавливает процессы обработки запросов на запись или чтение, которые всегда остаются более приоритетными. Чтобы операция полной балансировки не приводила к затруднению работы пользователей, ее следует выполнять в часы наименьшей загрузки кластера.

Для дальнейшей детализации методики построения кластерных систем больших объемов требуется разработка детальных алгоритмов работы каждой из подсистем, а также общего алгоритма работы узла кластера. Кроме того, требуется определить весовые коэффициенты параметров и математические модели для вычисления коэффициента загрузки узла.

ЛИТЕРАТУРА

1. *Пэйдж В. Использование Oracle 8/8i. – С.-Пб : Вильямс, 1999. – 632 с.*
2. *Чекмаров А. Новые технологии Windows 2000. – BHV, 1999. – 448 с.*
3. *Тэйт С. Windows 2000 для системного администратора. – С.-Пб : Питер, 2001. – 784 с.*
4. *Надточей А.Л. Применение кластерных систем для создания надежных высокоскоростных хранилищ данных больших объемов // Авиационно-космическая техника и технология. – Х.: НАКУ «ХАИ». – 2002. – С. 174 – 177.*

Поступила 20.03.2003

***НАДТОЧЕЙ Алексей Леонидович**, старший преподаватель кафедры компьютерных систем Национального аэрокосмического университета «ХАИ». В 1999 году окончил Национальный аэрокосмический университет «ХАИ». Область научных интересов – применение кластерных систем в информационных компьютерных сетях.*