



ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ,
ПАТЕНТАМ И ТОВАРНЫМ ЗНАКАМ

(51) МПК
G06F 17/30 (2006.01)
G06F 9/445 (2006.01)

(12) ОПИСАНИЕ ИЗОБРЕТЕНИЯ К ПАТЕНТУ

(21), (22) Заявка: 2005116849/09, 23.07.2004

(24) Дата начала отсчета срока действия патента:
23.07.2004

(30) Конвенционный приоритет:
15.12.2003 US 10/737,726

(43) Дата публикации заявки: 20.01.2006

(45) Опубликовано: 27.08.2009 Бюл. № 24

(56) Список документов, цитированных в отчете о
поиске: US 2003/0159136 A1, 21.08.2003. US 6425126
B1, 23.07.2002. US 6490722 B1, 03.12.2002. US
6000000 A, 07.12.1999. RU 2210803 C2,
20.08.2003.

(85) Дата перевода заявки РСТ на национальную
фазу: 01.06.2005

(86) Заявка РСТ:
US 2004/023965 (23.07.2004)

(87) Публикация РСТ:
WO 2005/060387 (07.07.2005)

Адрес для переписки:
129090, Москва, ул. Б.Спасская, 25, стр.3,
ООО "Юридическая фирма Городисский и
Партнеры", пат.пов. Ю.Д.Кузнецову,
рег.№ 595

(72) Автор(ы):

РОБЕРТС Джейсон (US),
МОХАММЕД Мазхар (US),
УИТТЕЛ Уолтер (US),
ШЕПАРД Марк (US)

(73) Патентообладатель(и):

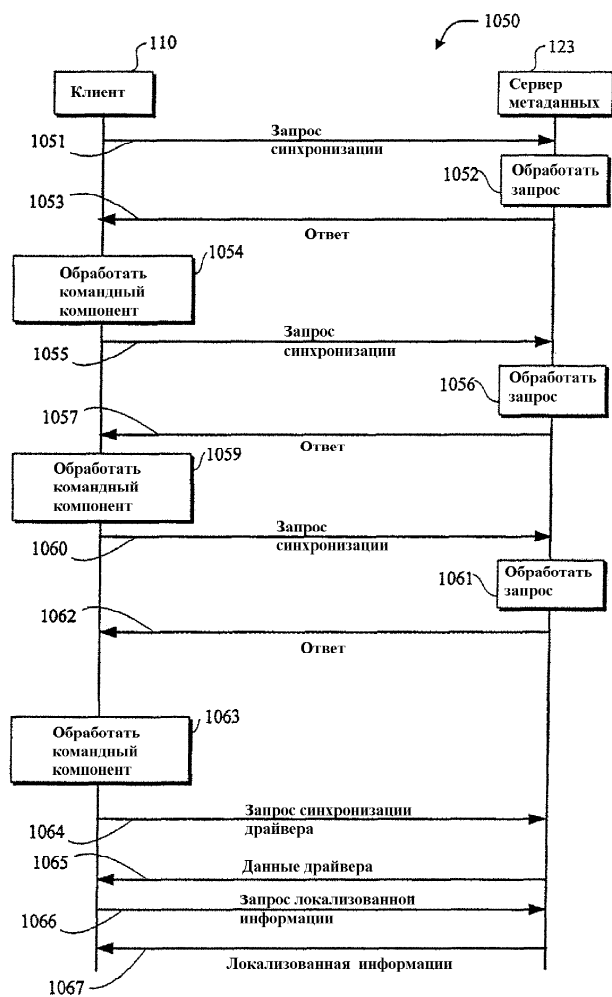
МАЙКРОСОФТ КОРПОРЕЙШН (US)

(54) СИСТЕМА И СПОСОБ ДЛЯ СЛУЖБЫ РАСПРОСТРАНЕНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

(57) Реферат:

Изобретение относится к системе и способу для управления обновлениями программного обеспечения. Техническим результатом является обеспечение возможности получать соответствующие обновления посредством эффективного использования уменьшенной полосы частот. В соответствии с настоящим изобретением служба обновления

программного обеспечения управляет доступом к обновлениям программного обеспечения, хранимым на серверах. В соответствии с другим вариантом служба обновления программного обеспечения осуществляет синхронизацию с клиентскими устройствами, чтобы идентифицировать применимые обновления. 5 н. и 17 з.п. ф-лы, 14 ил.



Фиг. 10



FEDERAL SERVICE
FOR INTELLECTUAL PROPERTY,
PATENTS AND TRADEMARKS

(51) Int. Cl.

G06F 17/30 (2006.01)*G06F 9/445* (2006.01)**(12) ABSTRACT OF INVENTION**(21), (22) Application: **2005116849/09, 23.07.2004**(24) Effective date for property rights:
23.07.2004(30) Priority:
15.12.2003 US 10/737,726(43) Application published: **20.01.2006**(45) Date of publication: **27.08.2009 Bull. 24**(85) Commencement of national phase: **01.06.2005**(86) PCT application:
US 2004/023965 (23.07.2004)(87) PCT publication:
WO 2005/060387 (07.07.2005)

Mail address:
**129090, Moskva, ul. B.Spasskaja, 25, str.3, OOO
"Juridicheskaja firma Gorodisskij i Partnery",
pat.pov. Ju.D.Kuznetsovu, reg.№ 595**

(72) Inventor(s):

**ROBERTS Dzhejson (US),
MOKhAMMED Mazkhar (US),
UITTEL Uolter (US),
ShEPARD Mark (US)**

(73) Proprietor(s):

MAJKROSOFT KORPOREJShN (US)**(54) SYSTEM AND METHOD FOR SOFTWARE DISTRIBUTION SERVICE**

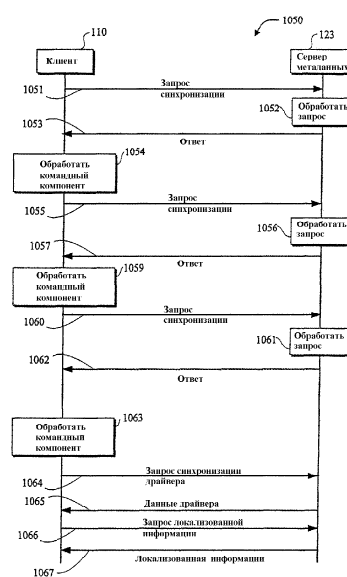
(57) Abstract:

FIELD: information technology.

SUBSTANCE: invention relates to a system and method for managing software update. In accordance with this invention, a software update service controls access to software updates stored on servers. In another version the software update service effects synchronisation with client devices so as to identify suitable updates.

EFFECT: possibility of obtaining suitable updates through efficient use of narrow frequency band.

21 cl, 14 dwg



Фиг. 10

ОБЛАСТЬ ТЕХНИКИ, К КОТОРОЙ ОТНОСИТСЯ ИЗОБРЕТЕНИЕ

Настоящее изобретение, в целом, относится к программному обеспечению и вычислительным сетям и, более конкретно, настоящее изобретение относится к системе и способу, предназначенным для управления и обмена обновлениями программного обеспечения.

ПРЕДШЕСТВУЮЩИЙ УРОВЕНЬ ТЕХНИКИ

Большинство коммерчески доступных программных продуктов подвергаются повторяющейся процедуре модификации, чтобы восстанавливать или совершенствовать функциональные возможности и/или функции. Каждая версия программного продукта или компонента может требовать добавления новых файлов и/или замены имеющихся файлов файлами более новых версий. Как только изготовитель выделил проблему программного продукта и создал реализацию решения проблемы, требуется поместить это устранение ошибки в обновление, и сделать обновление широкодоступным заказчиком. У изготовителей программного обеспечения есть деловая заинтересованность в распространении обновлений программного обеспечения заказчиком настолько быстро и безаварийно, насколько возможно.

Сеть Интернет предоставляет заказчикам важный канал для получения (самых) последних обновлений программных продуктов. Неограниченно возрастающее использование Интернет создало общее ожидание заказчиками того, что программные продукты и обновления должны поставлять в оперативном режиме, чтобы осуществлять загрузку по сети. В области интересов изготовителей программных средств находится также продвижение использования Интернет для распространения обновлений, поскольку это уменьшает их затраты и дает возможность заказчикам получать устранение ошибок для выявленной проблемы, как только устранение ошибок сделано доступным для загрузки. Сайты изготовителей в сети Интернет могут быть разработаны, чтобы сделать очень простым обнаружение и определение местоположения файлов обновления для применения. Технические аспекты загрузки файла по большей части исчезли из поля зрения пользователя, и теперь обычно обрабатываются операционной системой.

В традиционном подходе изготовитель программного обеспечения создает обновление программного обеспечения в виде "пакета" для загрузки. Такой пакет является обычно саморазворачивающимся исполняемым файлом вместе с установочной программой и каждым из обновленных файлов программного продукта, вложенным и сжатым, чтобы уменьшить пакет. Размер пакета обычно является суммой размеров каждого сжатого измененного файла, плюс размер непосредственно программы извлечения. При исполнении пакет извлекает каждый из содержащихся файлов во временное местоположение, затем запускает установочную программу, чтобы установить каждый файл на надлежащее местоположение в системном каталоге. Файлы, которые поставляют в сжатой форме, декомпрессируют по мере того, как их устанавливают. Любой существующий файл с одним и тем же наименованием в одном и том же местоположении будет просто записан заново в соответствии с файлом замены.

Даже при том, что Интернет делает возможным широкое и быстрое распространение программных обновлений, ограниченная ширина полосы передачи по сети вызвала сложности. Абсолютные (чистые) размеры приложений стандартного программного обеспечения вызвали то, что загружаемые объемы обновлений стали необоснованно большими. Обычно набор устранений ошибок для множества

проблем (ошибок) программного продукта группируют в обновление. Если изготовитель обновляет программный продукт на постоянной основе (систематически), то загружаемый объем пакета обновления будет продолжать увеличиваться, поскольку изготовитель не может не включать в состав файлы, исходя из предположения, что пользователь уже имеет данные файлы из более ранних обновлений. Поскольку пакет обновления объединяет некоторое количество полных файлов, он может быть весьма большим, даже если файлы сжаты. Иногда даже при самых быстрых модемных соединениях эффективность использования полосы частот для загрузки снижается.

Аспект больших затрат времени, соответствующий традиционному процессу обработки, загрузки по сети, конечно, является нежелательным. В некоторых случаях заказчики оплачивают расходы дальней (междугородной) связи или времени соединения в течение таких загрузок файлов. Любые сокращения времени соединения будут сокращать прямые денежные расходы для этих заказчиков. Изготовители обычно также имеют некоторые ощутимые затраты, относящиеся к объемам загрузок, которые они поставляют, поэтому сокращение объемов может дать им также прямые денежные выгоды. Сокращение объемов загрузок увеличит пропускную способность доступной им сети, позволяя им обслуживать большее количество заказчиков с помощью имеющегося оборудования сетевого сервера.

Значительное время, которое требуется, чтобы загрузить по сети большое (по размеру) обновление, также делает процесс загрузки более уязвимым к различным сложностям сетевого соединения. Имеется ряд причин, почему сеанс Интернет может быть разъединен преждевременно, включающие помехи в телефонной линии, сигналы ожидания запроса, и непреднамеренные команды. Некоторые поставщики услуг Интернет применяют ограничение времени соединения, ограничивая время, в течение которого пользователь может быть подсоединенным к сети в одном сеансе. Если пользователь загружает большой файл в момент, когда сетевое соединение отключают, ему, или ей, вероятно, придется осуществить повторный запуск. Большинство обычных операционных систем и протоколов передачи файлов не позволяют возобновить передачу файла, поэтому любое временное продвижение будет потеряно, и передачу нужно будет повторно запустить. Возможности для отказа столь многочисленны, что многие пользователи находят почти невозможным получение обновления в оперативном режиме. Если размер пакета обновления является слишком большим, то пользователи возможно никогда не смогут загрузить его полностью.

Одна попытка сократить размер обновлений программного обеспечения и повысить эффективность использования полосы частот относится к использованию небольших корректирующих изменений, т.е. дельта-корректировок, или двоичных корректировок. Специалист в соответствующей области техники оценит, что дельта-корректировка соответствует специальному программному коду, который изменяет существующий файл при исполнении вычислительным устройством. Поскольку дельта-корректировка включает в себя специальный программный код, то необходима уникальная дельта-корректировка для каждой уникальной версии файла. В применении к обновлениям программного обеспечения служба обновления программного обеспечения может передавать меньшую по размеру обновленную дельта-корректировку вместо передачи полного, обновленного файла. Обновленную дельта-корректировку затем используют, чтобы модифицировать существующий файл в обновленный файл.

Несмотря на то, что обновляемые дельта-корректировки могут потенциально сократить объем данных, требуемых для того, чтобы обновить файлы, современные подходы к осуществлению дельта-корректировок являются несовершенными в управлении выбором применимых дельта-файлов в ситуациях, в которых существует большое количество версий файла. Поскольку необходима уникальная дельта-корректировка для каждой версии файла, то типичные системы обновления программного обеспечения могут обычно требовать, чтобы сотни, если не тысячи, уникальных дельта-корректировок соответствовали каждой уникальной версии файла. В одном подходе некоторые службы обновления, поддерживающие осуществление дельта-корректировок, передают все возможные дельта-корректировки на клиентское вычислительное устройство. Однако этот подход обычно увеличивает объем данных, требуемых для осуществления обновления программного обеспечения, поскольку увеличивается количество возможных обновляемых дельта-корректировок. Соответственно, количество потенциально применимых дельта-корректировок может быстро возрасти к такому же размеру, как у полного обновленного файла. В другом подходе сетевая служба обновления программного обеспечения просматривает клиентское устройство, чтобы выбрать, какая дельта-корректировка может быть применима для каждого клиентского устройства. Хотя это уменьшает объем передаваемой информации дельта-корректировки, это требует дополнительного логического блока в службе обновления программного обеспечения для просмотра клиентских устройств и выбора применимой дельта-корректировки. Использование дополнительного логического блока увеличивает системные ресурсы, которые должны быть обеспечены службой. Дополнительно, данный подход обычно препятствует использованию кэширования сети такого, какое обычно выполняют традиционные Web-серверы.

В дополнение к вышеописанным недостаткам, существующие системы являются неспособными поставлять некоторые типы обновлений программного обеспечения, например, драйверов, аппаратного обеспечения. Как известно в данной области техники, обновления специального программного обеспечения, например, такие, которые применяют к драйверам аппаратного обеспечения, являются трудными для поставки пользователям в крупномасштабном распространении, поскольку большая часть обновлений специального программного обеспечения будет работать только на клиентских компьютерах со специфическими аппаратными средствами. В большинстве случаев, например, если клиентский компьютер получает несовместимую обновленную версию драйвера аппаратного обеспечения, установка обновленной версии драйвера может вызвать фатальную ошибку или даже препятствовать действию компьютера.

Как понятно из предшествующего, имеется потребность в системе и способе усовершенствованного обмена обновлениями программного обеспечения между сервером и рядом клиентов. Кроме того, имеется потребность в системе и способе обновления программного обеспечения, которые имеют усовершенствованные механизмы предоставления услуг обновления на целевые конкретные типы клиентов при доставке специальных обновлений.

КРАТКОЕ ИЗЛОЖЕНИЕ СУЩНОСТИ ИЗОБРЕТЕНИЯ

Настоящее изобретение посвящено системе и способу, предназначенным для управления обновлениями программного обеспечения. Более конкретно, настоящее изобретение направлено на систему и способ, предназначенные для содействия выбору и осуществлению обновлений программного обеспечения, уменьшая при этом

ресурсы полосы частот и обработки, требуемые для того, чтобы выбирать и осуществлять обновления программного обеспечения. В соответствии с аспектом настоящего изобретения, служба обновления программного обеспечения управляет доступом к обновлениям программного обеспечения, хранмым на серверах. В соответствии с другим аспектом, служба обновления программного обеспечения осуществляет синхронизацию с клиентскими устройствами, чтобы выявить (идентифицировать) применимые обновления.

В соответствии с аспектом настоящего изобретения, обеспечен способ обмена информацией между клиентским процессом вычислительного устройства и серверным процессом в системе распределенной обработки данных. В соответствии со способом клиентский процесс выдает запрос синхронизации. Запрос синхронизации включает в себя идентификатор установленного обновления программного обеспечения, если клиентское вычислительное устройство хранит установленное обновление программного обеспечения. Серверный процесс принимает запрос синхронизации и выбирает дополнительное обновление программного обеспечения, если запрос синхронизации содержит идентификатор по меньшей мере одного обновления программного обеспечения. Выбор дополнительного обновления программного обеспечения зависит от выполнения предварительного условия, заданного в дополнительном обновлении программного обеспечения. Предварительное условие требует, чтобы запрос синхронизации включал в себя идентификатор по меньшей мере одного конкретного обновления программного обеспечения. Серверный процесс выбирает обновление программного обеспечения, относящееся к первому уровню, которое имеет нулевое значение в качестве предварительного условия, если он определяет, что запрос синхронизации не содержит идентификатор по меньшей мере одного обновления программного обеспечения. Серверный процесс выдает командный компонент выбранного обновления программного обеспечения. Клиентский процесс принимает командный компонент выбранного обновления программного обеспечения и сохраняет командные компоненты выбранного обновления программного обеспечения в клиентском вычислительном устройстве в качестве установленного обновления программного обеспечения, если клиентское вычислительное устройство содержит по меньшей мере один компонент, который выполняет условие правила применимости, хранимого в выбранном обновлении программного обеспечения.

В соответствии с другим аспектом настоящего изобретения обеспечен способ обмена информацией между клиентским процессом вычислительного устройства и серверным процессом в системе распределенной обработки данных. В соответствии с этим способом клиентский процесс выдает запрос прав или запрос авторизации. Запрос авторизации содержит идентификатор связанного с целевой группой модуля авторизации клиента. Серверный процесс принимает запрос авторизации и выдает элемент используемых сервером идентификационных данных, или серверный элемент идентификационных данных («cookie» куки-данные), идентифицирующий целевую группу, если серверный процесс обнаруживает наличие серверного программного модуля, связанного с целевой группой. Клиентский процесс принимает серверные куки-данные, идентифицирующие целевую группу, и выдает запрос на обновление по меньшей мере для одного обновления программного обеспечения, причем запрос содержит серверные куки-данные. Серверный процесс принимает запрос на обновление и выдает обновление программного обеспечения, если обновление программного обеспечения является связанным с целевой группой,

идентифицированной в серверных куки-данных. Клиентский процесс принимает обновление программного обеспечения, если обновление программного обеспечения является связанным с целевой группой, идентифицированной в серверных куки-данных.

5 В соответствии с дополнительным аспектом настоящего изобретения обеспечена система обмена и управления обновлениями программного обеспечения. Система включает в себя первую базу данных, хранящую набор обновлений программного обеспечения. Индивидуальное обновление программного обеспечения, хранимое в
10 базе данных, может включать в себя поле командных данных, содержащее правило применимости, задающее по меньшей мере одно условие применимости, включая нулевое значение. Командный компонент содержит набор предварительных условий, идентифицирующих одно или несколько обновлений программного обеспечения, требуемых для установки индивидуального обновления программного обеспечения,
15 включая нулевое значение. Индивидуальное обновление программного обеспечения, хранимое в базе данных, может также включать в себя поле локализованных данных, содержащее текстовые данные, описывающие общую информацию, относящуюся к индивидуальному обновлению программного обеспечения. Дополнительно,
20 индивидуальное обновление программного обеспечения, хранимое в базе данных, может включать в себя компонент данных, который идентифицирует по меньшей мере один файл потока данных, связанный с индивидуальным обновлением программного обеспечения.

Система может также включать в себя сервер авторизации для санкционирования
25 доступа к обновлениям программного обеспечения, хранимым в первой базе данных. Сервер авторизации выдает серверные куки-данные для разрешения клиентскому вычислительному устройству доступа к индивидуальному обновлению программного обеспечения, если клиентское вычислительное устройство содержит модуль
30 авторизации, указывающий, что клиентское вычислительное устройство является связанным с целевой группой. Серверные куки-данные содержат идентификатор целевой группы. Дополнительно, серверные куки-данные передают на клиентское вычислительное устройство для сохранения серверных куки-данных.

Система может дополнительно включать в себя сервер метаданных для поставки
35 обновлений на клиентское вычислительное устройство. Обновление программного обеспечения передают на клиентское вычислительное устройство, если обновление программного обеспечения является связанным с целевой группой, причем сервер метаданных соединен с возможностью информационного обмена с первой базой
40 данных. Сервер метаданных может получать запрос синхронизации. Запрос синхронизации включает в себя идентификатор установленного обновления программного обеспечения, если клиентское вычислительное устройство хранит установленное обновление программного обеспечения. Запрос синхронизации
45 включает в себя серверные куки-данные. Сервер метаданных может также выбирать дополнительное обновление программного обеспечения для передачи на клиентское вычислительное устройство, если запрос синхронизации содержит идентификатор по меньшей мере одного установленного обновления программного обеспечения. Выбор
50 дополнительного обновления программного обеспечения зависит от выполнения предварительного условия, заданного в дополнительном обновлении программного обеспечения. Предварительное условие требует, чтобы запрос синхронизации включал в себя идентификатор по меньшей мере одного установленного обновления программного обеспечения. Дополнительное обновление программного обеспечения

является связанным с целевой группой, идентифицированной в серверных куки-данных. Сервер метаданных также выбирает для передачи на клиентское вычислительное устройство обновление программного обеспечения, соответствующее первому уровню, если запрос синхронизации не содержит идентификатор по меньшей мере одного установленного обновления программного обеспечения. Обновления программного обеспечения первого уровня не включают в себя предварительно условие. Сервер метаданных может передавать командный компонент выбранного обновления программного обеспечения на клиентское вычислительное устройство для сохранения и получать запрос на локализованные данные, связанные с выбранным обновлением программного обеспечения. Дополнительно сервер метаданных может передавать на клиентское вычислительное устройство локализованные данные, связанные с выбранным обновлением программного обеспечения. Локализованные данные дают возможность клиентскому вычислительному устройству формировать запрос загрузки для приема компонента данных выбранного обновления программного обеспечения.

Система может дополнительно включать в себя сервер загрузки, коммуникативно соединенный с первой базой данных. Сервер загрузки может получать запрос загрузки от клиентского вычислительного устройства. Запрос загрузки идентифицирует компонент данных выбранного обновления программного обеспечения. Сервер загрузки может передавать компонент данных выбранного обновления программного обеспечения на клиентское вычислительное устройство.

В соответствии с другим аспектом настоящего изобретения обеспечена система обновления данных на клиентском вычислительном устройстве. Система включает в себя первую базу данных, хранящую набор дельта-корректировок. Система также включает в себя серверный компьютер, коммуникативно соединенный с первой базой данных, и клиентское вычислительное устройство. Этот сервер может получать выборку одного или нескольких доступных обновлений программного обеспечения для корректировки одного или нескольких файлов, установленных на клиентском вычислительном устройстве, саморазворачивающийся файл, идентифицирующий набор доступных дельта-корректировок для корректировки по меньшей мере одной версии установленного файла и инвентаризационной информации относительно одного или нескольких файлов, установленных на клиентском вычислительном устройстве. Сервер также может выбирать одну или несколько применимых дельта-корректировок, чтобы осуществить выбранные программные обновления. Выбор одной или нескольких применимых корректировок соответствует отображению саморазворачивающегося файла, идентифицирующего набор доступных дельта-корректировок, на инвентаризационную информацию относительно одного или нескольких файлов, установленных на клиентском вычислительном устройстве. Сервер может дополнительно передавать запрос одной или нескольких выбранных дельта-корректировок.

В соответствии со следующим аспектом настоящего изобретения обеспечена система, предназначенная для передачи обновлений программного обеспечения на клиентское вычислительное устройство. Система включает в себя первый компонент, хранящий набор серверных модулей авторизации. Индивидуальный серверный модуль авторизации идентифицирует целевую группу. Система дополнительно включает в себя серверный компьютер, коммуникативно соединенный с первым компонентом. Этот сервер может получать запрос авторизации от клиентского вычислительного устройства. Запрос авторизации содержит идентификатор сохраняемого на

клиентском вычислительном устройстве модуля авторизации клиента. Сервер может дополнительно определять, связано ли клиентское вычислительное устройство с целевой группой, если модуль авторизации клиента указывает, что клиентское вычислительное устройства является связанным с целевой группой. Сервер может
 5 дополнительно передавать серверные куки-данные от службы обновления программного обеспечения на клиентское вычислительное устройство, если клиентское вычислительное устройство является связанным с целевой группой. Серверные куки-данные идентифицируют целевую группу. Сервер также может
 10 получать запрос по меньшей мере на одно обновление программного обеспечения, хранимое службой обновления программного обеспечения, причем запрос содержит серверные куки-данные. В ответ на прием запроса сервер может определить, связано ли обновление программного обеспечения с целевой группой, идентифицированной в серверных куки-данных. Дополнительно, если обновление программного обеспечения
 15 связано с целевой группой, идентифицированной в серверных куки-данных, сервер может передавать обновление программного обеспечения от службы обновления программного обеспечения на клиентское вычислительное устройство.

В соответствии с очередным аспектом настоящего изобретения, обеспечен
 20 машиночитаемый носитель, содержащий хранимую на нем структуру данных (для представления) индивидуального обновления программного обеспечения. Машиночитаемый носитель включает в себя поле командных данных, содержащее правило применимости, задающее по меньшей мере одно условие применимости, включая нулевое значение. Командный компонент содержит набор предварительных
 25 условий, которые идентифицируют одно или несколько обновлений программного обеспечения, которые необходимы для установки индивидуального обновления программного обеспечения, включая нулевое значение. Машиночитаемый носитель также может включать в себя поле локализованных данных, содержащее текстовые
 30 данные, описывающие общую информацию, связанную с индивидуальным обновлением программного обеспечения. Машиночитаемый носитель может дополнительно включать в себя компонент данных, который идентифицирует по меньшей мере один файл потока данных, связанный с индивидуальным обновлением программного обеспечения.

35 КРАТКОЕ ОПИСАНИЕ ЧЕРТЕЖЕЙ

Предшествующие аспекты и многие из сопутствующих преимуществ данного изобретения будут оценены и лучше поняты при ссылке на последующее подробное описание, рассматриваемое вместе с сопроводительными чертежами, на которых:

40 Фиг.1 - блок-схема системы обновления программного обеспечения, включающей в себя клиентский компьютер и службу обновления, поставляющую обновляемое программное обеспечение, в соответствии с настоящим изобретением;

Фиг.2 - блок-схема системы обновления программного обеспечения согласно Фиг.1, иллюстрирующая аутентификацию клиентского вычислительного устройства с
 45 помощью службы обновления в соответствии с настоящим изобретением;

Фиг.3 - блок-схема системы обновления программного обеспечения согласно Фиг.1, иллюстрирующая синхронизацию доступных обновлений между клиентским вычислительным устройством и службой обновления в соответствии с настоящим изобретением;

50 Фиг.4 - блок-схема системы обновления программного обеспечения согласно Фиг.1, иллюстрирующая передачу информации об обновлении программного обеспечения на клиентское вычислительное устройство от службы обновления в соответствии с

настоящим изобретением;

Фиг.5 - блок-схема системы обновления программного обеспечения согласно Фиг.1, иллюстрирующая обработку и выбор информации об обновлении, осуществляемые клиентским вычислительным устройством, в соответствии с настоящим изобретением;

Фиг.6 - блок-схема системы обновления программного обеспечения согласно Фиг.1, иллюстрирующая объединение дельта-корректировок и установку обновленных файлов посредством клиентского вычислительного устройства в соответствии с настоящим изобретением;

Фиг.7 - иллюстративная схема последовательности операций программы обновления программного обеспечения, осуществляемой клиентским вычислительным устройством и службой обновления для идентификации обновлений программного обеспечения, доступных для установки на клиентском вычислительном устройстве, в соответствии с настоящим изобретением;

Фиг.8 - схема протокола программы аутентификации для обеспечения избирательного доступа к обновлениям, хранимым службой обновления, в соответствии с настоящим изобретением;

Фиг.9 - блок-схема примерного набора обновлений программного обеспечения, иллюстрирующая процедуру авторизации, в соответствии с настоящим изобретением;

Фиг.10 - схема протокола программы синхронизации для передачи выбранной группы обновлений программного обеспечения от службы обновления программного обеспечения на клиентское вычислительное устройство, в соответствии с настоящим изобретением;

Фиг.11 - блок-схема, иллюстрирующая примерный раздел графического пользовательского интерфейса для визуального представления перечня обновлений программного обеспечения, которые являются доступными индивидуальному клиентскому вычислительному устройству, в соответствии с настоящим изобретением;

Фиг.12А и 12В - иллюстрируют подпрограмму 1200 обработки обновления программного обеспечения, осуществляемую клиентским вычислительным устройством 110 для извлечения и установки запрошенного программного обеспечения, в соответствии с настоящим изобретением; и

Фиг.13 - схема последовательности операций, иллюстративная для подпрограммы, осуществляемой клиентским вычислительным устройством для обновления базового установочного компонента в соответствии с настоящим изобретением.

ПОДРОБНОЕ ОПИСАНИЕ ПРЕДПОЧТИТЕЛЬНОГО ВАРИАНТА ОСУЩЕСТВЛЕНИЯ

Описываемое в целом настоящее изобретение относится к системе и способу, предназначенным для управления обновлениями программного обеспечения. Более конкретно, настоящее изобретение направлено на систему и способ, содействующие выбору и осуществлению обновлений программного обеспечения, уменьшая при этом полосу частот и ресурсы обработки, необходимые, чтобы выбирать и осуществлять обновления программного обеспечения. В соответствии с настоящим изобретением обновления программного обеспечения могут соответствовать обновлениям для конкретных программных приложений или операционных систем. Дополнительно, обновления программного обеспечения могут включать в себя программные драйверы или обновления программно-аппаратного обеспечения, например, системной БСВВ (BIOS). В соответствии с аспектом настоящего изобретения обеспечена системная и компонентная архитектура для обработки обновлений программного обеспечения. В соответствии с другим аспектом настоящего

изобретения обеспечены протокол обновления и интерфейс для содействия санкционированию доступа (авторизации) и синхронизации клиентских устройств с помощью службы обновления. В соответствии с дополнительным аспектом настоящего изобретения обеспечены способ обновления компонента установки и различных установленных файлов с использованием дельта-корректировки. Специалист в соответствующей области техники оценит, тем не менее, что дополнительные аспекты настоящего изобретения также могут быть обеспечены в настоящей заявке. Дополнительно специалист в соответствующей области техники оценит, что каждый обозначенный аспект может рассматриваться индивидуально или в качестве части общего аспекта изобретения.

Система 100 обновления программного обеспечения на Фиг.1 является иллюстративной блок-схемой системы 100 обновления программного обеспечения в соответствии с настоящим изобретением. Описываемая в целом, система 100 обновления программного обеспечения может содержать одно или несколько клиентских вычислительных устройств 110, службу 120 обновления и внешнего поставщика 130 обновлений. Описываемая в целом, служба 120 обновления хранит и управляет распространением обновлений программного обеспечения, которые передают на клиентское вычислительное устройство 110 и устанавливают на нем. Обновления программного обеспечения могут быть поставлены службой 120 обновления или произвольным количеством внешних поставщиков 130 обновлений.

Клиентское вычислительное устройство 110, служба 120 обновления и внешний поставщик 130 обновлений с помощью электроники осуществляют обмен информацией по сети 101. Сеть может быть локальной сетью (ЛВС, LAN) или более крупной сетью такой, как глобальная сеть (ГВС, WAN) или Интернет. При использовании общеизвестного программного обеспечения система 100 обновления программного обеспечения может быть настроена с возможностью обмена документами, командами и другими известными типами информации, между клиентским вычислительным устройством 110 и серверами 121, 122, 123 и 124 службы 120 обновления. Как будет оценено специалистами в данной области техники и другими, система 100 обновления программного обеспечения, показанная на Фиг.1, является упрощенным примером одной подходящей системы для осуществления настоящего изобретения и что настоящее изобретение не является ограниченным данным примером.

Как описано более подробно ниже, один вариант осуществления службы 120 обновления содержит несколько серверов. Как показано на Фиг.1, служба 120 обновления включает в себя сервер 121 обновления для управления всеми процессами службы 120 обновления и координирования процессов серверов 121, 122, 123 и 124 службы 120 обновления. Сервер 122 авторизации формирует куки-данные авторизации согласно запрошенному клиентом и, в свою очередь, куки-данные авторизации используют, чтобы сформировать серверные куки-данные, которые разрешают клиентским компьютерам осуществлять доступ к обновлениям, поставляемым службой 120 обновления. Сервер 123 метаданных обеспечивает общую информацию, относящуюся к обновлениям, поставляемым службой 120 обновления. Сервер 123 метаданных дает возможность системе, соответствующей настоящему изобретению, идентифицировать конкретные обновления для конкретного типа клиентского компьютера или конкретной группы клиентских компьютеров. Сервер 124 загрузки обеспечивает один или несколько программных компонентов для доставки файлов данных, связанных с обновлениями программного обеспечения, поставляемыми

службой 120 обновления.

Внешний поставщик 130 обновлений может включать в себя один или несколько серверов, которые распространяют обновления программного обеспечения. Внешний поставщик 130 обновлений может быть соотнесен (связан) с объектом, который 5 предоставляет программное обеспечение, обновления программного обеспечения, или другие данные, подлежащие распространению на группы клиентских компьютеров. Например, внешний поставщик 130 обновлений может быть связан со сторонним разработчиком программного обеспечения, желающим использовать службу 120 10 обновления, чтобы распространять обновления для одного или нескольких программных приложений. В другом примере внешний поставщик 130 обновлений может быть связан со службой 120 обновления программного обеспечения.

Клиентское вычислительное устройство 110 может быть любым вычислительным устройством, которое хранит и исполняет программные приложения 114. Клиентское 15 вычислительное устройство 110 может быть создано из любого одного из ряда различных продуктов вычислительной техники, включая в них, но не ограничиваясь таковыми: персональные компьютеры (ПК, РС), персональные цифровые ассистенты (PDA), мобильные телефоны, двусторонние устройства персонального вызова, или пейджеры, и т.д. Как будет оценено средними специалистами в данной 20 области техники или другими, архитектура клиентского вычислительного устройства 110 может принимать любую подходящую форму. Например, клиентское вычислительное устройство 110 может включать в себя сетевой интерфейс для обеспечения обмена информацией с сетью 101. Сетевой интерфейс может быть 25 настроен с возможностью использования с любым соединением сети проводной или беспроводной связи, и может быть использован с помощью любого подходящего протокола управления передачей таким, как протокол ТСР/ІР. Кроме того, клиентское вычислительное устройство 110 может включать в себя блок обработки, или 30 процессор, устройство отображения, и блок запоминающего устройства. Блок запоминающего устройства может хранить программный код, необходимый для функционирования клиентского вычислительного устройства 110, такой как операционная система 116. В дополнение, блок запоминающего устройства хранит компонент 112 управления обновлением, предназначенный для управления и 35 исполнения процессов согласно настоящему изобретению.

Система 100 обновления программного обеспечения хранит программы, реализованные программно, которые при исполнении осуществляют настоящее изобретение. При исполнении система 100 обновления программного обеспечения 40 хранит, управляет и выборочно (избирательно) передает обновления программного обеспечения. Как описано более полно ниже, среди многих других преимуществ настоящее изобретение обеспечивает механизм определения и выбора целевых групп клиентских вычислительных устройств, которые могут быть избранными для приема обновлений программного обеспечения. Настоящее изобретение также обеспечивает 45 усовершенствованный механизм осуществления загрузки файлов данных, связанных с обновлениями программного обеспечения.

Для целей иллюстрирования настоящего изобретения предусмотрено подробное описание действующего примера настоящего изобретения. В описании действующего 50 (рабочего) примера сделана ссылка на обновления программного обеспечения, которые могут относиться к конкретному обновлению программного приложения, например, к обновлению версии 6.0 программы воспроизведения мультимедийных данных на версию 7.0 программы воспроизведения мультимедийных данных. Как

будет оценено средними специалистами в данной области техники, такое обновление программного обеспечения может включать в себя передачу и установку ряда файлов данных, связанных с обновлением программного обеспечения. Таким образом, для целей иллюстрирования настоящего изобретения различие между обновлением программного обеспечения и отдельным файлом данных, содержащим обновление программного обеспечения.

Ниже со ссылкой на Фиг.2-6 описано иллюстративное взаимодействие между компонентами системы 100 обновления программного обеспечения, осуществляемое, чтобы обновить один или несколько файлов на клиентском вычислительном устройстве 110. Со ссылкой на Фиг.2, службу обновления программного обеспечения инициируют посредством передачи информации об обновлении программного обеспечения одним или несколькими внешними поставщиками 130 обновлений. Как описано выше, внешние поставщики 130 обновлений могут быть связаны с системой 100 обновления программного обеспечения. В качестве альтернативы, информация об обновлении программного обеспечения может быть передана сторонними внешними поставщиками 130 обновлений. В иллюстративном варианте осуществления настоящего изобретения информация об обновлении программного обеспечения может включать в себя программный код, используемый для обновления файла, программный код, используемый для замены файла, различные правила для определения применимости обновлений программного обеспечения, и/или визуального представления информации, описывающей обновление программного обеспечения. Передача информации об обновлении программного обеспечения может быть завершена в любое время и необязательно должна быть одновременной с инициированием взаимодействий других проиллюстрированных компонентов обновления программного обеспечения.

После приема от внешнего поставщика 130 обновлений информации об обновлении программного обеспечения служба 120 обновления формирует один или несколько блоков данных, чтобы содействовать передаче информации обновления. Данные могут включать в себя файл хранилища корректировок, который соответствует набору программных дельта-корректировок для обновления различных версий файла. Данные могут также включать в себя декларацию (именование) хранилища корректировок, которая соответствует индексу, отображающему конкретные версии файлов на соответствующую дельта-корректировку, находящуюся в файле хранилища корректировок. Данные дополнительно могут включать в себя саморазворачивающийся файл, соответствующий информации, которую агент обновления будет использовать для того, чтобы запрашивать и устанавливать данные обновления конкретного программного обеспечения, как описано более подробно ниже. Специалист в данной области техники оценит, что формирование файла хранилища корректировок, декларации хранилища корректировок и саморазворачивающихся файлов может быть выполнимым в любой момент времени и необязательно должно быть одновременным с взаимодействиями других проиллюстрированных компонентов.

Чтобы инициировать передачу клиентам информации об обновлении программного обеспечения, клиентское вычислительное устройство 110 инициирует к службе 120 обновления запрос аутентификации. В иллюстративном варианте осуществления настоящего изобретения запрос аутентификации соответствует взаимодействию по протоколу обновления между клиентским вычислительным устройством 110 и службой 120 обновления, который описан более подробно ниже.

После завершения аутентификации служба 120 обновления передает куки-данные аутентификации на клиентское вычислительное устройство 110. На Фиг.3 показано, что прошедшее аутентификацию клиентское вычислительное устройство 120 затем инициирует синхронизацию доступных обновлений с сервером 120 обновлений. В иллюстративном варианте осуществления настоящего изобретения запрос синхронизации также соответствует взаимодействию по протоколу обновления между клиентским вычислительным устройством 110 и службой 120 обновления, который описан более подробно ниже. После завершения синхронизации клиентское вычислительное устройство 110 принимает информацию всех применимых обновлений программного обеспечения и информацию, описывающую обновления. Однако в иллюстративном варианте осуществления настоящего изобретения не загружается программный код, чтобы создать экземпляр обновления.

Со ссылкой на Фиг.3, в некоторый момент в течение процесса обновления принимают выборку обновлений, подлежащих установке. В иллюстративном варианте осуществления настоящего изобретения пользователю может быть представлена информация об обновлении программного обеспечения, принятая в течение синхронизации, и предложено выбрать подходящее обновление. В качестве альтернативы, клиентское вычислительное устройство 110 может быть настроено таким образом, чтобы автоматически выбирать все применимые обновления программного обеспечения. Дополнительно, клиентское вычислительное устройство 110 также может содержать некоторые правила, которые позволяют ему автоматически выбирать поднабор доступных обновлений программного обеспечения. Дополнительно пользователь может инициировать выборку обновления посредством обмена информацией со службой 120 обновления, например, через web-страницу Интернет.

Со ссылкой на Фиг.4, компонент 112 управления обновлением создает агент 118 обновления на клиентском вычислительном устройстве 110, если агент обновления не является уже существующим. Агент 118 обновления затем запрашивает передачу пакета информации об обновлении программного обеспечения, например, саморазворачивающегося файла. Агент 118 обновления принимает саморазворачивающийся файл и выполняет какие-либо (имеющиеся) обновления к программе установки (инсталлятору), как описано ниже. Дополнительно агент 118 обновления может запросить любую отсутствующую или разрушенную информацию от службы 120 обновления.

Со ссылкой на Фиг.5, как только агент 118 обновления принимает пакет информации об обновлении программного обеспечения, агент 118 обновления выполняет инвентаризацию файлов, которые установлены на клиентском вычислительном устройстве 110. На основании сравнения инвентаризационной информации и пакета информации об обновлении программного обеспечения агент 118 обновления определяет, какая дельта-корректировка или другая информация об обновлении необходима для выполнения выбранных обновлений. Агент 118 обновления затем передает запрос на конкретные дельта-обновления. В одном варианте осуществления настоящего изобретения запрос на обновления программного обеспечения может соответствовать передаваемому через непосредственное сетевое соединение непосредственному запросу, который обозначен как ручное обновление. В другом варианте осуществления настоящего изобретения запрос на обновления программного обеспечения может быть фоновым запросом, который передают, не требуя явного действия пользователя. Этот вариант

осуществления обозначен как автоматическое обновление.

В иллюстративном варианте осуществления настоящего изобретения, если обновление программного обеспечения соответствует дельта-корректировке, то агент 118 обновления передает на службу 120 обновления запрос, который
 5 идентифицирует конкретную дельта-корректировку, идентифицированную согласно декларации хранилища корректировок. В качестве альтернативы, в случае, когда дельта-корректировка является недоступной или если несколько дельта-корректировок потерпели неудачу, агент 118 обновления может инициировать
 10 программу нейтрализации ошибок, или восстановления. Программа восстановления может включать в себя запрос на передачу полной копии полностью обновленного файла из файла хранилища корректировок. Программа восстановления также может включать в себя запрос на передачу в отдельном пакете полной копии полностью обновленного файла.

В иллюстративном варианте осуществления настоящего изобретения сервер 124 загрузки, относящийся к службе 120 обновления, может непосредственно обрабатывать запрос на обновление программного обеспечения от агента 118 обновления. В качестве альтернативы, запрос также может быть обработан любым
 20 количеством дополнительных внешних серверов загрузки, например, традиционными Web-серверами, которые тоже приняли запрошенные дельта-корректировки обновления от службы 120 обновления. Например, корпорация может использовать внутренний сервер для обновления клиентских устройств. Дополнительно, запрос может быть обработан внешними серверами загрузки, в
 25 которых некоторые, или все дельта-корректировки обновления являются кэшированными при обработке предыдущих запросов. Соответственно, в этом варианте осуществления загрузка может быть распределенной на нескольких дополнительных серверах загрузки, способных обслуживать запросы данных, соответствующие протоколу передачи гипертекстовых файлов ("HTTP").
 30

Со ссылкой на Фиг.6, как только принята информация об обновлении программного обеспечения, агент 118 обновления объединяет дельта-корректировку с установленным файлом, чтобы сформировать обновленный файл. Дополнительно, агент 118 обновления может подтверждать правильность того, успешно ли программа
 35 объединения обновила соответствующий файл. Как описано выше, если правильность дельта-корректировки не может быть подтверждена, агент 118 обновления может запрашивать дельта-корректировку вновь или запрашивать полностью обновленный файл после некоторого количества неудач. Как только агент 118 обновления получает
 40 подтвержденный и обновленный файл, файл устанавливается на клиентском вычислительном устройстве 110.

На Фиг.7 показана схема последовательности операций программы 700 обработки обновления программного обеспечения, иллюстрирующая взаимодействие между клиентским вычислительным устройством 110 и службой 120 обновления
 45 программного обеспечения в соответствии с настоящим изобретением. На этапе 702 служба 120 обновления программного обеспечения осуществляет авторизацию доступа к клиентскому компьютеру 110. В иллюстративном варианте осуществления настоящего изобретения, авторизация доступа к клиентскому компьютеру может
 50 включать в себя формирование серверных куки-данных для разрешения доступа к обновлениям программного обеспечения, которые связаны с конкретной группой компьютеров. Более подробное объяснение процесса авторизации описано со ссылками на Фиг.8.

На этапе 704 клиентский компьютер 110 и служба 120 обновления программного обеспечения синхронизируют информацию обновления. В иллюстративном варианте осуществления настоящего изобретения служба 120 обновления программного обеспечения передает на клиентское вычислительное устройство 110 метаданные, описывающие конкретные обновления программного обеспечения. Метаданные содержат информацию, описывающую доступные обновления программного обеспечения, чтобы дать возможность пользователю выбрать одно или несколько обновлений для установки. Более подробное описание процесса синхронизации будет приведено ниже со ссылками на Фиг.9 и 10. На этапе 706 клиентское вычислительное устройство 110 получает выборку применимых обновлений для загрузки. В иллюстративном варианте осуществления настоящего изобретения выборка применимых обновлений может соответствовать использованию некоторого количества уникальных пользовательских интерфейсов, чтобы содействовать выборкам пользователя. Выборка пользовательских интерфейсов описана более подробно со ссылками на Фиг.11.

На этапе 708 клиентское вычислительное устройство 110 обрабатывает пользовательскую выборку применимых обновлений программного обеспечения и связывается со службой 120 обновления программного обеспечения, чтобы запросить информацию конкретного обновления. В иллюстративном варианте осуществления настоящего изобретения клиентское вычислительное устройство 110 выбирает и запрашивает одну или несколько применимых дельта-корректировок обновлений. Агент 118 обновления, имеющийся на клиентском вычислительном устройстве 110, затем может обрабатывать запрошенные данные, чтобы осуществить выбранное обновление программного обеспечения. На этапе 710 программа 700 завершается.

Со ссылкой на Фиг.8 описана схема 800 протокола авторизации доступа к клиентскому вычислительному устройству 110 и соответствующая этапу 702 (Фиг.7). В иллюстративном варианте осуществления настоящего изобретения служба 120 обновления программного обеспечения использует механизм расширяемого определения целевого объекта, чтобы управлять доступом клиентского вычислительного устройства 110 к обновлениям и другому программному обеспечению. Служба 120 обновления программного обеспечения включает в себя механизм, который связывает конкретные обновления программного обеспечения с одной или несколькими целевыми группами клиентских вычислительных устройств 110. Например, служба 120 обновления программного обеспечения может ограничивать доступ относительно обновления конкретного драйвера аппаратного обеспечения до конкретной (торговой) марки клиентских вычислительных устройств 110, содержащих конкретное аппаратное устройство. В таком примере служба 120 обновления программного обеспечения может определять целевую группу клиентских вычислительных устройств 110, имеющих конкретную (торговую) марку и конкретное аппаратное устройство, и ограничивать передачу конкретной загрузки программного обеспечения на целевую группу.

В иллюстративном варианте осуществления настоящего изобретения механизму расширяемого определения целевого объекта содействует использование программных компонентов ("подключаемые дополнения авторизации"), которые задают принадлежность клиентского вычислительного устройства к одной или нескольким целевым группам. Наличие "дополнения авторизации" на клиентском вычислительном устройстве 110 задает, принадлежит ли клиентское вычислительное устройство к специфической целевой группе, относящейся к дополнению авторизации.

Целевая группа, например, может включать в себя все компьютеры, имеющие действующий идентификационный номер программного продукта ("ИДПП") для конкретного программного приложения. В таком примере, как описано подробно ниже в отношении Фиг.8, дополнение 826 авторизации может быть установлено на клиенте, чтобы считывать ИДПП из модуля памяти клиентского вычислительного устройства и передавать полученный ИДПП на соответствующее ИДПП-дополнение 829 сервера. Соответствующее ИДПП-дополнение, также обозначаемое при этом как блок 829 проверки достоверности ИДПП, использует один или несколько способов, чтобы определить, является ли полученный ИДПП действительным. Как только определено, что ИДПП, хранимый на клиентском вычислительном устройстве 110, является действительным, сервер формирует серверные куки-данные, которые указывают, что клиентское вычислительное устройство 110 является элементом целевой группы, имеющей действительный ИДПП. В другом примере целевая группа может включать в себя клиентские вычислительные устройства, которые обозначены как компьютеры бета-версии (опытной эксплуатации).

В иллюстративном варианте осуществления настоящего изобретения сервер 122 авторизации, относящийся к службе 120 обновления программного обеспечения, содержит ряд серверных дополнений авторизации, который задает набор целевых групп клиентских вычислительных устройств, которые сервер авторизации будет распознавать. Каждое серверное дополнение авторизации содержит компоненты для обмена данными с соответствующим клиентским дополнением авторизации, хранимым на клиентском вычислительном устройстве 110. Подобным образом, каждое клиентское вычислительное устройство 110 включает в себя одно или несколько клиентских дополнений авторизации, которые идентифицируют целевые группы, к которым принадлежит клиент. В иллюстративном варианте осуществления настоящего изобретения клиентские дополнения авторизации могут быть установлены на каждом клиентском вычислительном устройстве в течение установки или обновления программного приложения, например, установки или обновления операционной системы. Дополнительно, серверные дополнения авторизации могут быть динамически установлены или удалены администратором, желающим управлять доступом к обновлениям программного обеспечения. Дополнения авторизации, хранимые на клиентском вычислительном устройстве 110, и сервер 122 авторизации могут быть фактическим программным дополнением, или дополнения авторизации могут быть жестко запрограммированы в динамически связываемые библиотеки.

Как показано на Фиг.8, сервер 122 авторизации содержит три примерных серверных дополнения авторизации: (1) первое серверное дополнение 828 авторизации, задающее целевую группу, которая включает в себя все компьютеры (далее в документе "целевая группа Все Компьютеры"); (2) второе серверное дополнение 829 авторизации, задающее целевую группу, которая включает в себя компьютеры, имеющие действительный ИДПП (в дальнейшем "целевая группа ИДПП"); и (3) третье серверное дополнение 830 авторизации, задающее целевую группу, которая включает в себя компьютеры бета-тестирования (в дальнейшем "целевая группа Бета"). Также на Фиг.8 показано, что клиентское вычислительное устройство 110 содержит два клиентских дополнения авторизации: (1) первое клиентское дополнение 825 авторизации, указывающее, что клиентское вычислительное устройство 110 является элементом целевой группы «Все Компьютеры»; и (2) второе клиентское дополнение 826 авторизации, указывающее, что клиентское вычислительное

устройство 110 является элементом целевой группы ИДПП. В этом примере клиентское вычислительное устройство 110 не содержит дополнение авторизации, указывающее, что оно является элементом целевой группы Бета. Как будет оценено средними специалистами в данной области техники, каждое из клиентских дополнений 825 и 826 авторизации может быть настроено, чтобы исполнять одну или несколько функций на клиентском вычислительном устройстве 110, чтобы способствовать процессу проверки достоверности. Например, второе клиентское дополнение 826 авторизации может быть настроено для исследования запоминающего устройства клиентского вычислительного устройства 110 для того, чтобы проверить или получить ИДПП для установленного программного приложения.

Как показано на Фиг.8, подпрограмма 702 авторизации начинается тогда, когда клиентское вычислительное устройство 110 передает запрос 803 конфигурации на сервер 122 авторизации. В иллюстративном варианте осуществления настоящего изобретения, запрос 803 конфигурации формируют на основании произвольного подходящего программного компонента, настроенного с возможностью получать информацию, описывающую дополнения авторизации, хранимые на сервере 122 авторизации. Как будет оценено специалистами в данной области техники, запрос 803 конфигурации может использовать известный метод, обозначаемый как "GetConfig" («Получить конфигурацию»). В ответ на прием запроса 803 конфигурации сервер 122 авторизации передает ответ 804 о конфигурации, который включает в себя информацию, идентифицирующую все дополнения авторизации, хранимые на сервере 122 авторизации. В одном варианте осуществления ответ 804 о конфигурации включает в себя таблицу из строк, которая идентифицирует и описывает все дополнения авторизации, хранимые на сервере 122 авторизации. В данном примере ответ 804 о конфигурации включает в себя информацию, которая идентифицирует первое серверное дополнение 828 авторизации, второе серверное дополнение 829 авторизации, и третье серверное дополнение 830 авторизации.

На этапе 805 клиентское вычислительное устройство 110 формирует один или несколько куки-данных авторизации в ответ на прием ответа 804 о конфигурации. В ходе обработки согласно этапу 805 клиентское вычислительное устройство 110 формирует куки-данные авторизации для каждой пары согласующихся (соответствующих) клиентских и серверных дополнений авторизации. Таким образом, в данном примере, первое клиентское дополнение 825 авторизации формирует первые куки-данные авторизации, связанные с целевой группой «Все Компьютеры», поскольку и первое клиентское дополнение 825 авторизации, и первое серверное дополнение 828 авторизации являются связанными с целевой группой «Все Компьютеры». Кроме того, второе клиентское дополнение 826 авторизации формирует вторые куки-данные авторизации, связанные с целевой группой ИДПП, поскольку второе клиентское дополнение 826 авторизации и второе серверное дополнение 829 авторизации оба связаны с целевой группой ИДПП. Третьи куки-данные авторизации не формируют, поскольку клиентское вычислительное устройство 110 не имеет дополнения авторизации, указывающего, что оно является элементом целевой группы Бета.

Как будет также оценено специалистами в данной области техники, одно осуществление операций обработки на этапе 805 может включать в себя использование общеизвестного программного метода, обозначаемого в данной области техники как "GetAuthCookie" («Получить элемент данных авторизации»). Будет также оценено, что формирование каждого куки-данных авторизации может

включать в себя дополнительную обработку. Например, второе клиентское дополнение 826 авторизации может быть настроено с возможностью проверки информации, хранимой в клиентском системном реестре, чтобы извлекать ИДПП и включать ИДПП в куки-данные авторизации. В других примерах, процесс, соответствующий этапу 805, может включать в себя процессы для взаимодействия с другими компьютерами или устройствами. Например, клиентское дополнение авторизации может обмениваться информацией с устройством таким, как звуковая плата, сканирующее устройство, видео плата, и т.д., для того, чтобы получить тип и модель устройства. В других неограничивающих примерах клиентское дополнение авторизации может обмениваться информацией с устройством (обеспечения) безопасности, например, считывателем отпечатков пальцев, чтобы получить информацию, описывающую пользователя.

В целом клиентское дополнение авторизации может считывать информацию о конфигурации из любого компонента клиентского вычислительного устройства 110 или любого другого вычислительного устройства, связанного для обмена информацией с клиентским вычислительным устройством 110. В других примерах клиентское дополнение авторизации может быть настроено с возможностью использования одного или нескольких общедоступных или скрытых (секретных) программных интерфейсов приложений (API), чтобы собирать и шифровать информацию от клиента, который будет проверен посредством соответствующего серверного дополнения. В таких примерах, дополнение 826 проверки достоверности ИДПП использует секретный API, чтобы зашифровать ИДПП клиента для передачи зашифрованного ИДПП на сервер для расшифровывания и проверки достоверности. В других вариантах осуществления другие клиентские дополнения авторизации могут использовать результаты биометрических измерений, например, считыватель отпечатков пальцев или "отпечаток голоса", для того, чтобы создать куки-данные авторизации, которые подлежат передаче на сервер для проверки достоверности. В следующем примере клиентское дополнение авторизации может вызывать услугу сети Web или любую другую услугу, чтобы передать мандат (или учетную запись) авторизации или любой другой тип данных на сервер 122 авторизации.

В иллюстративном варианте осуществления настоящего изобретения каждые куки-данные авторизации включают в себя строку, которая идентифицирует соответствующую целевую группу. Например, строка может указывать, что конкретные куки-данные авторизации связаны с целевой группой ИДПП. Каждые куки-данные авторизации также включают в себя раздел данных для передачи данных между клиентом и сервером. Например, куки-данные авторизации, связанные с целевой группой ИДПП, могут иметь раздел данных, который содержит фактический ИДПП. Как будет оценено средними специалистами в данной области техники, раздел данных может содержать любой тип данных, которые сохраняют в любом формате, например, массив байтов. Например, если дополнения на клиенте и сервере требуют передач открытых и секретных ключей, такие данные могут быть зашифрованы в разделе данных одного или нескольких куки-данных авторизации.

Как только клиентское вычислительное устройство 110 сформирует куки-данные авторизации для каждой пары соответствующих клиентских и серверных дополнений авторизации, клиентское вычислительное устройство передает сформированные куки-данные авторизации на сервер 122 авторизации. Как показано на Фиг.8, клиентское вычислительное устройство 110 передает куки-данные авторизации в куки-запросе 806. Куки-запрос 806 включает в себя любой подходящий формат для

передачи массива куки-данных авторизации, сформированных в процессе, соответствующем этапу 805. Одно осуществление этой части метода 702 авторизации может включать в себя использование общеизвестного программного метода, обозначаемого в данной области техники как "GetCookie" (Получить элемент

идентификационных данных).
 В одном варианте осуществления куки-запрос 806 также включает в себя другие куки-данные сервера авторизации, сохраняемые на запоминающем устройстве клиентского вычислительного устройства 110. Как станет понятно из последующего описания, запоминающее устройство клиентского вычислительного устройства 110 может хранить предыдущие куки-данные сервера авторизации, которые были созданы в ходе предыдущих исполнений программы 700 авторизации. Предусматривая хранимые куки-данные сервера авторизации в куки-запросе 806, клиентское вычислительное устройство 110 будет способно поддерживать его собственные привилегии доступа, которые предоставлялись в ходе предыдущих исполнений подпрограммы 702 авторизации. В данном примере, поскольку нет куки-данных сервера авторизации, хранимых на клиенте, куки-запрос 806 включает в себя первые куки-данные авторизации, связанные с «целевой группой Все Компьютеры» и вторые куки-данные авторизации, связанные с целевой группой ИДПП.

Затем, как показано на этапе 807, в ответ на прием куки-запроса 806 сервер 122 авторизации формирует серверные куки-данные. В одном варианте осуществления для каждого принятых куки-данных авторизации осуществляют вызов соответствующего серверного дополнения авторизации, чтобы сформировать серверные куки-данные. Эти серверные куки-данные, сформированные посредством каждого серверного дополнения авторизации, включают в себя идентификатор для каждой целевой группы, идентифицированной в принятых куки-данных авторизации. В данном примере, поскольку куки-запрос 806 включает в себя первые куки-данные авторизации, связанные с «целевой группой Все Компьютеры», и вторые куки-данные авторизации, связанные с целевой группой ИДПП, сервер 122 авторизации формирует серверные куки-данные, содержащие идентификатор для этих соответствующих целевых групп. На сервере 122 авторизации серверные куки-данные затем объединяют с предыдущими серверными куки-данными, если предыдущие серверные куки-данные приняты в куки-запросе 806, чтобы сформировать новые серверные куки-данные. В одном варианте осуществления новые серверные куки-данные шифруют с помощью способа шифрования с открытым ключом, такого, как тройной DES (стандарт шифрования данных).

В иллюстративном варианте осуществления настоящего изобретения серверные куки-данные могут включать в себя зашифрованную информацию, которая идентифицирует одну или несколько соответственных (ему) целевых групп. Кроме того, серверные куки-данные могут включать в себя данные о сроке действия, которые сохраняют и в формате открытого текста, и в зашифрованном формате. Данные о сроке действия, хранимые в формате открытого текста, использует клиентское вычислительное устройство 110 для текущего контроля срока действия серверных куки-данных. Данные о сроке действия, хранимые в зашифрованном формате, использует служба 120 обновления программного обеспечения, чтобы определить, полномочно ли клиентское вычислительное устройство 110 получать обновления, связанные с конкретной целевой группой. В одном варианте осуществления, данные о сроке действия серверных куки-данных применяют ко всем целевым группам, идентифицированным в серверных куки-данных. В качестве

альтернативы, или в дополнение ко времени срока действия, применяемому к взятым в целом серверным куки-данным, серверные куки-данные могут включать в себя множество данных о сроках действия, каждые из которых может применяться к индивидуальным целевым группам. Как будет оценено средними специалистами в данной области техники, каждые серверные куки-данные могут включать в себя дополнительные данные. Например, серверные куки-данные могут быть настроены с возможностью сохранять информацию о состоянии клиента, такую как отметка времени последнего исполнения подпрограммы 702 авторизации.

Как только сформированы куки-данные 809 сервера авторизации передают из сервера 122 авторизации на клиентское вычислительное устройство 110. Далее, как показано на этапе 811, серверные куки-данные затем сохраняют в запоминающем устройстве клиентского вычислительного устройства 110. В случае, когда клиентское вычислительное устройство 110 определяет, что срок действия по меньшей мере одного компонента из серверных куки-данных истек, клиентское вычислительное устройство может повторно исполнить метод 702 авторизации, чтобы получить новые серверные куки-данные. Как упомянуто выше, в каждом последующем исполнении метода 702 авторизации клиентское вычислительное устройство 110 может передавать свои сохраненные серверные куки-данные на сервер 122 авторизации в куки-запросе 806. В одном варианте осуществления клиент не должен посылать запрос 803, до тех пор, пока сервер не сообщит клиенту, что конфигурация сервера изменилась, то есть было добавлено новое дополнение авторизации.

В соответствии с другим аспектом настоящего изобретения служба 120 обновления программного обеспечения может обеспечивать подпрограмму синхронизации для осуществления синхронизации обновляемой информации между сервером 123 метаданных и клиентским вычислительным устройством 110. При помощи однозначной (уникальной) иерархии обновлений программного обеспечения подпрограмма синхронизации может эффективно идентифицировать конкретные обновления, которые применяют к конкретному клиентскому вычислительному устройству. Кроме того, при помощи серверных куки-данных, сформированных в подпрограмме 702 авторизации, подпрограмма синхронизации может избирательно предоставлять доступ к обновлениям, связанным с конкретными целевыми группами.

В соответствии с иллюстративным вариантом осуществления настоящего изобретения каждое обновление программного обеспечения включает в себя три компонента: (1) командный компонент; (2) компонент локализованных данных; и (3) компонент данных. Как будет оценено средним специалистом в данной области техники, каждое обновление может иметь один или несколько вышеописанных компонентов. Например, обновление может содержать командный компонент, компонент локализованных данных и компонент потока данных. В другом примере обновление может содержать только командный компонент для проверки одного или нескольких условий клиентского вычислительного устройства. Различные компоненты обновлений программного обеспечения описаны более подробно ниже.

Описываемый в целом командный компонент содержит два подкомпонента: (1) правило применимости, которое задает одно или несколько условий применимости, которые должны быть проверены клиентским вычислительным устройством 110; и (2) набор предварительных условий, который идентифицирует одно или несколько обновлений, которые необходимы для надлежащей установки индивидуального обновления. Как описано ниже, правило применимости может задавать набор условий применимости, относящихся к компьютеру, и каждое из условий может быть связано с

другими условиями с использованием произвольных логических операторов.

Например, командный компонент может включать в себя правило применимости, чтобы определять, содержит ли компьютер установленную конкретную версию Windows®. Как также описано ниже, набор предварительных условий может идентифицировать одно или несколько обновлений, которые должны быть предварительно установленными. Например, как описано более подробно ниже со ссылкой на Фиг.9, индивидуальное обновление может содержать предварительное условие, которое содержит перечень других обновлений, необходимых для надлежащей установки индивидуального обновления. В других примерах, как также показано на Фиг. 9, набор предварительных условий может включать в себя использование логических операторов, чтобы задавать более сложные правила для предварительных условий.

Командный компонент также содержит код, например, логический флажок, который указывает, есть ли другие обновления, которые зависят от конкретного обновления. Для иллюстративных целей обновление рассматривают являющимся обновлением «ЛИСТ» (LEAF), если нет других обновлений, которые зависят от данного обновления. Логический флажок, который используют, чтобы указать, является ли обновление LEAF-обновлением, динамически модифицирует сервер 123 метаданных по мере того, как соответствующие обновления добавляют или удаляют.

Компонент локализованных данных для каждого обновления включает в себя общую информацию, описывающую обновление. Например, компонент локализованных данных может включать в себя информацию, описывающую функциональные возможности и эффект обновления. Компонент локализованных данных также может включать в себя текстовое описание процедур установки обновления. Кроме того, компонент локализованных данных может включать в себя любые другие данные или информацию, относящуюся к обновлению. Например, локализованные данные могут указывать, что обновление является высокоприоритетным обновлением. В другом примере локализованные данные могут обеспечивать специальные установочные сообщения, например, сообщение, которое указывает, что обновление не может быть установлено вместе с другими обновлениями программного обеспечения. Локализованная информация может быть в формате, который предусматривает визуальное представление содержащейся в нем информации пользователю.

Компонент данных для каждого обновления включает в себя один или несколько потоков двоичных данных, соответствующих обновлению. В одном варианте осуществления компонент данных каждого обновления может быть связан с одним или несколькими файлами данных, такими как исполняемый файл, документ, подключаемая библиотека, и т.д. Как описано более подробно ниже, каждое обновление может быть связано с комбинацией файлов данных, каждый из которых содействует фактическому обновлению, установке или модификации программного обеспечения, используемого клиентом. Например, установке обновления может содействовать использование одиночного CAB-файла-кабинет, включающего всю информацию, необходимую для выполнения выбранного обновления. В качестве альтернативы, установке обновления может содействовать использование набора индивидуальных обновлений, используемых для обновления одного или нескольких файлов, хранимых на клиентском вычислительном устройстве.

В иллюстративном варианте осуществления настоящего изобретения обновления программного обеспечения могут быть скомпонованы в иерархию, которая дает

возможность управляемого распространения обновлений программного обеспечения. Описываемая в целом иерархия обновлений задает отношения между обновлениями и конкретно указывает, какие обновления зависят от других обновлений. Для иллюстративных целей предусмотрен и показан на Фиг.9 примерный набор обновлений. Как показано, иерархия примерных обновлений 900 включает в себя базовый набор обновлений 901, второй набор обновлений 902 и третий набор обновлений 903. В целом, каждое обновление в базовом наборе обновлений 901 не имеет предварительного условия, которое требует установки других обновлений. Однако шестое обновление 921 содержит предварительное условие, которое требует установки первого обновления 911, второго обновления 912 и третьего обновления 913. Седьмое обновление 922 содержит предварительное условие, которое требует установки четвертого обновления 914. Восьмое обновление 931 содержит предварительное условие, которое требует установки шестого обновления 921 и пятого обновления 915. Следовательно, восьмое обновление 931 также требует установки первого обновления 911, второго обновления 912, и третьего обновления 913. С целями иллюстрирования настоящего изобретения задано, что все обновления примерного набора обновлений 900 являются (все) связанными с целевой группой «Все Компьютеры» и целевой группой ИДПП.

На Фиг.9 также показано, и как описано более подробно ниже, что каждое обновление содержит правило применимости, которое задает условия применимости для установки обновления. Например, первое обновление 911 требует установки англоязычной версии операционной системы. Второе обновление 912 требует установки Windows® XP вместе с версией 1 служебного пакета SP1. В другом примере шестое обновление 921 требует установки корректировки программного обеспечения, обозначенной как XP PATCH1. Соответственно, клиентское вычислительное устройство 110 не будет устанавливать обновления, если правила применимости не были удовлетворены.

На Фиг.9 также показана секция командного компонента, которая указывает, есть ли другие обновления, которые зависят от конкретного обновления. Ссылку на эту секцию обозначают как LEAF, которая может быть в формате логического значения, указывающего, что обновление программного обеспечения является последним обновлением последовательности связанных обновлений. С целями иллюстрирования изобретения конкретное обновление является LEAF-обновлением, если нет другого обновления, которое вносит в перечень конкретное обновление в качестве предварительного условия. Как показано на Фиг.9, только седьмое обновление 922 и восьмое обновление 931 являются двумя LEAF-обновлениями.

На Фиг.10 проиллюстрирована схема протокола для подпрограммы 704 синхронизации (Фиг.7), определенной в соответствии с настоящим изобретением. Описываемая в целом подпрограмма 704 синхронизации выборочно передает командные компоненты некоторых обновлений между клиентским вычислительным устройством 110 и сервером, например сервером 123 метаданных, для того чтобы идентифицировать обновления, которые могут быть применимы к клиентскому вычислительному устройству 110. Как показано на Фиг.10, чтобы инициировать обновление, клиентское вычислительное устройство 110 сначала обрабатывает установленные обновления и передает запрос 1051 синхронизации на сервер 123 метаданных, запрашивая одно или несколько обновлений, доступных для клиента. В ответ на прием запроса 1051 синхронизации, сервер 123 метаданных возвращает набор обновлений на клиентское вычислительное устройство 110. Как понятно из

нижеследующего описания, клиентское вычислительное устройство 110 обрабатывает локально-хранимые данные прежде передачи запроса 1051 синхронизации.

Клиентское вычислительное устройство 110 обрабатывает командные компоненты каждого принятого обновления, чтобы определить, может ли быть удовлетворено
 5 условие применимости, заданное в правилах применимости. Если условие применимости, заданное в индивидуальном обновлении, удовлетворено, то для целей иллюстрирования подпрограммы 1050 синхронизации индивидуальное обновление является "устанавливаемым", и устанавливаемое обновление сохраняют в первом
 10 компоненте клиентского кэша обновления. С другой стороны, если условие, заданное в индивидуальном обновлении, не удовлетворено, то индивидуальное обновление рассматривают являющимся "неуспешным", и неуспешное обновление сохраняют во втором компоненте клиентского кэша обновления. В данном описании подпрограммы 1050 синхронизации, если обновление установлено, то можно
 15 предполагать, что предварительные условия и условия правил применимости были удовлетворены. Для целей описания данной подпрограммы установка обновления не обязательно означает, что файлы данных, связанные с обновлением, являются фактически установленными в клиентском вычислительном устройстве 110.

В одном варианте осуществления могут быть использованы два компонента клиентского кэша обновления, чтобы принятые обновления разделить на категории. Первый компонент используют для сохранения устанавливаемых, не-LEAF-обновлений; и второй компонент используют для сохранения всех остальных обновлений, принятых клиентом, то есть обновлений, которые не были установлены.
 25 Второй компонент кэша обновления также включает в себя хранилище всех LEAF-обновлений. Как описано более подробно ниже, обновления, хранимые в кэше обновления, могут быть переданы на сервер 123 метаданных и обработаны им для того, чтобы идентифицировать другие связанные обновления, которые являются
 30 доступными для установки на клиентском вычислительном устройстве 110.

Возвращаясь к Фиг.10, ниже описаны подробности запроса синхронизации, которые проиллюстрированы в качестве позиций 1051, 1055, и 1060. Как будет оценено средними специалистами в данной области техники, запрос синхронизации может быть
 35 инициирован одним из многих различных устройств, процессов, приложений, инициированных пользователем команд, запрашивающих обновление. Запрос синхронизации может быть инициирован пользователем, запрашивающим перечень обновлений, автоматическим обновлением, инициированным клиентским агентом, или любым другим программным компонентом, запросившим информацию от
 40 сервера 123 метаданных или службы 120 обновления. В одном варианте осуществления запрос синхронизации включает в себя серверные куки-данные авторизации, например серверные куки-данные авторизации, сформированные программой 702 авторизации. Использование серверных куки-данных позволяет серверу определять, является ли клиент элементом одной или нескольких целевых
 45 групп.

Каждый запрос синхронизации также может включать в себя идентификаторы для каждого обновления, хранимого в клиентском кэше обновления. Более конкретно, если одно или несколько обновлений хранятся в кэше обновления, то запрос
 50 синхронизации включает в себя первый компонент, содержащий идентификаторы для устанавливаемых неLEAF-обновлений; и второй компонент, содержащий идентификаторы всех остальных обновлений таких, как LEAF-обновления, неуспешных обновлений и других обновлений, которые не являются установленными.

Идентификаторы обновлений могут быть в любом формате, включающем, но не ограниченным таковым, массив целых чисел. В качестве альтернативы, если нет обновлений, сохраненных в клиентском кэше обновления, запрос синхронизации формируют без идентификатора обновления. В случае, когда запрос синхронизации формируют без идентификатора обновления, запрос синхронизации предусматривает указание того, что клиентское вычислительное устройство 110 не имеет кэшированных обновлений.

Как показано на Фиг.10, первый запрос 1051 синхронизации передают из клиентского вычислительного устройства 110 на сервер 123 метаданных. В данном примере клиентский кэш обновления не будет содержать какие-либо обновления, задавая, что это является первым исполнением способа. Таким образом, первый запрос 1051 синхронизации не содержит идентификатор кэшированного обновления. В ответ на прием запроса синхронизации, как показано на этапе 1052, сервер 123 метаданных определяет, содержит ли запрос синхронизации по меньшей мере один идентификатор обновления. Если определяют, что запрос синхронизации не включает в себя идентификатор обновления, то сервер 123 метаданных осуществляет ответ, выбирая обновление первого уровня для передачи на клиентское вычислительное устройство 110. Как описано выше, обновления первого уровня могут включать в себя любые обновления, которые не имеют предварительных условий, идентифицирующих другие обновления.

В качестве альтернативы, если определено, что запрос синхронизации содержит по меньшей мере один идентификатор обновления, сервер 123 метаданных исследует предварительные условия хранимых сервером обновлений, чтобы выбрать дополнительные обновления для доставки клиенту. В одном варианте осуществления сервер 123 метаданных выбирает обновления, имеющие выполненные предварительные условия. При проверке предварительных условий сервер использует обновления из первого компонента запроса синхронизации, который включает в себя идентификаторы neLEAF-обновлений, установленных на клиенте.

В дополнение к выбору обновлений, имеющих выполненные предварительные условия, сервер также использует обновления, идентифицированные во втором компоненте запроса синхронизации, чтобы отфильтровать выбранные обновления. Более конкретно, удаляемые установленные обновления, LEAF-обновления и неуспешные обновления, идентифицированные во втором компоненте запроса синхронизации, используют, чтобы отфильтровать одно или несколько выбранных обновлений. Этот признак настоящего изобретения позволяет системе и способу настоящего изобретения избегать многократных передач обновлений, хранимых на сервере 123 метаданных.

Возвращаясь к данному примеру, поскольку первый запрос 1051 синхронизации не включает в себя идентификатор обновления, сервер 123 метаданных выбирает базовый уровень 901 обновлений для передачи на клиентское вычислительное устройство 110. Со ссылкой на примерный набор обновлений, показанный на Фиг.9, базовый уровень обновлений 901 включает в себя обновления, обозначенные как 911, 912, 913, 914, и 915.

В ходе обработки согласно этапу 1052 сервер 123 метаданных также исследует серверные куки-данные авторизации, содержащийся в запросе 1051 синхронизации, для того, чтобы идентифицировать целевые группы, которые связаны с клиентским вычислительным устройством 110. Сервер 123 метаданных также исследует целевые группы обновлений, выбранных в процессе согласно этапу 1052. Обработка согласно

этапу 1052 затем отфильтровывает все выбранные обновления, которые не являются связанными с целевой группой, идентифицированной в принятых куки-данных сервера авторизации. В данном примере, поскольку все выбранные обновления 911, 912, 913, 914, и 915 связаны с целевыми группами ИДПП и «Все Компьютеры», то все
 5 выбранные обновления посылают на клиентское вычислительное устройство 110.

Затем сервер 123 метаданных передает на клиентское вычислительное устройство 110 выбранные обновления в ответе 1053 (на запрос) синхронизации. В целом, каждый ответ синхронизации включает в себя командный компонент каждого
 10 обновления, посылаемого сервером 120. Таким образом, в данном примере, первый ответ 1053 синхронизации включает в себя командные компоненты для обновлений, обозначенных 911, 912, 913, 914, и 915. В одном варианте осуществления каждый ответ синхронизации не включает в себя компонент локализованных данных или компонент
 15 данных для каждого обновления.

Затем, как показано на этапе 1054, клиентское вычислительное устройство 110 обрабатывает командные компоненты каждого принятого обновления для того, чтобы определить, может ли быть удовлетворено условие, заданное в правилах применимости. Со ссылкой вновь на Фиг.9, клиентское вычислительное
 20 устройство 110 обрабатывает командные компоненты принятых обновлений 911-915. Для целей иллюстрирования настоящего изобретения в этом примере задано, что операционной системой клиентского вычислительного устройства 110 является установка системы англоязычной версии Windows® XP SP1. Также задано, что клиентским вычислительным устройством 110 является ПК Dell PC и рабочим
 25 процессором является 32-разрядный X86. Таким образом, в ходе обработки командных компонентов примерного набора обновлений клиентское вычислительное устройство 110 определит, что условие, заданное в первом обновлении 911, будет удовлетворено, поскольку компьютер содержит англоязычную OS. Условие, заданное
 30 во втором обновлении 912, будет удовлетворено, поскольку операционной системой является Windows® версии XP, SP1. Условие, заданное в третьем обновлении 913, будет удовлетворено, поскольку клиентское вычислительное устройство 110 задействует процессор X86. Условие, заданное в пятом обновлении 915, будет удовлетворено, поскольку клиентским вычислительным устройством 110 является ПК Dell PC. В
 35 качестве результата первое обновление 911, второе обновление 912, третье обновление 913 и пятое обновление 915 все сохранены в первом компоненте клиентского кэша обновления. Условие, заданное в четвертом обновлении 914, не будет удовлетворено, поскольку клиентское вычислительное устройство 110 не
 40 исполняет 64-разрядный процессор X86. Таким образом, четвертое обновление 914 рассматривают являющимся неуспешным обновлением и сохраняют во втором компоненте клиентского кэша обновления.

Возвращаясь к Фиг.10, в обработке согласно этапу 1054, клиентское вычислительное устройство 110 также определяет, необходим ли последующий запрос
 45 синхронизации. В одном варианте осуществления определяют, что последующий запрос синхронизации необходим, если по меньшей мере одно из принятых обновлений указывает, что оно не является LEAF-обновлением. В данном примере определяют, что последующий запрос синхронизации необходим, поскольку все
 50 принятые обновления не являются LEAF-обновлениями. Таким образом, клиентское вычислительное устройство 110 передает последующий запрос 1055 синхронизации на сервер 123 метаданных.

В качестве вышеприведенного итога запрос синхронизации включает в себя

идентификаторы для каждого обновления, сохраняемого в клиентском кэше обновления. Таким образом, в данном примере последующий запрос 855 синхронизации включает в себя первый компонент данных, указывающий, что первое обновление 711, второе обновление 712, третье обновление 713 и пятое обновление 715 являются устанавливаемыми на клиенте. Кроме того, последующий запрос 855 синхронизации включает в себя второй компонент данных, указывающий, что четвертое обновление 711 не является успешно установленным на клиенте.

В ответ на прием последующего запроса 1055 синхронизации, в качестве итога вышеприведенного, сервер 123 метаданных определяет, содержит ли последующий запрос 1055 синхронизации по меньшей мере один идентификатор обновления. Если определено, что последующий запрос синхронизации содержит по меньшей мере один идентификатор обновления, сервер 123 метаданных исследует предварительные условия всех сохраненных обновлений, чтобы выбрать дополнительные обновления для доставки на клиент.

Со ссылкой вновь на данный пример, в обработке согласно этапу 856 сервер 123 метаданных выберет шестое обновление 921, поскольку его предварительные условия удовлетворены. Более конкретно, как показано на Фиг.9, выбрано шестое обновление 921 для передачи на клиентское вычислительное устройство 110, поскольку его предварительное условие, которое требует установки первого обновления 711, второго обновления 912 и третьего обновления 913, выполнено. Седьмое обновление 922 и восьмое обновление 931 не будут выбраны для передачи на клиентское вычислительное устройство, поскольку их предварительные условия не являются выполненными. Более конкретно, запрос 1055 синхронизации не содержал идентификатор для четвертого обновления 914, которое является предварительным условием для седьмого обновления 922. Кроме того, запрос 855 синхронизации не содержал идентификатор для шестого обновления 921, которое является предварительным условием для восьмого обновления 931.

Возвращаясь к Фиг.10, подпрограмма 1050 синхронизации продолжается передачей выбранных обновлений в последующем ответе 1057 от сервера 123 метаданных на клиентское вычислительное устройство 110. Со ссылкой вновь на данный пример, последующий ответ 1057, который будет включать в себя информацию, относящуюся к шестому обновлению 721, передают на клиентское вычислительное устройство 110 в последующем ответе 1057.

После приема последующего ответа 1057 клиентское вычислительное устройство 110 обрабатывает командные компоненты последующего ответа 857. Подобно обработке согласно этапу 854, клиентское вычислительное устройство 110 обрабатывает командные компоненты каждого принятого обновления, чтобы определить, удовлетворено ли условие применимости, заданное в правилах применимости. В данном примере, если задано, что установлена (ОС) XP RATCH1 в клиентском вычислительном устройстве, шестое обновление 921 рассматривают как устанавливаемое и обновление записывают в кэш обновления клиентского вычислительного устройства 110. Так как шестое обновление 921 не является LEAF-обновлением, то клиентское вычислительное устройство 110 посылает еще один запрос 1060 синхронизации, который включает в себя все обновления, хранимые в первом и втором компоненте клиентского кэша обновления. Запрос 1060 синхронизации также включает в себя куки-данные сервера авторизации.

В данном примере при использовании вышеописанной обработки посредством сервера 123 метаданных запрос 1060 синхронизации обрабатывают на этапе 1061, на

котором сервер выбирает восьмое обновление 931. Восьмое обновление 931 выбирают, поскольку запрос 1060 синхронизации указывает, что пятое и шестое обновления 915 и 921 являются установленными в клиентском вычислительном устройстве 110. Учитывая, что восьмое обновление 931 связано с теми же целевыми группами, которые идентифицированы в куки-данных сервера авторизации, командный компонент восьмого обновления 931 передают на клиентское вычислительное устройство 110 в другом ответе 1062. Восьмое обновление 931 затем обрабатывают на этапе 1063 способом, подобным процессу согласно этапам 1054 и 1059. Так как все принятые обновления ответа 1062 являются LEAF-обновлениями, последующий запрос синхронизации не посылают обратно на сервер 123 метаданных.

В клиентском вычислительном устройстве 110 после того, как определено, что все принятые обновления не являются LEAF-обновлениями, или если нет обновлений, принятых в ответе 1062, подпрограмма 1050 синхронизации от клиентского вычислительного устройства 110 передает запрос 1064 синхронизации драйвера на сервер 123 метаданных. Как будет оценено средним специалистом в данной области техники, запрос 1064 синхронизации драйвера может включать в себя информацию, описывающую все аппаратное обеспечение, установленное в клиентском вычислительном устройстве 110, и информацию, описывающую установленное программное обеспечение. Подобно предшествующим запросам синхронизации (1051, 1055 и 1060) программного обеспечения запрос 1064 синхронизации драйвера может передавать установленные обновления на сервер. Кроме того, все обновления драйвера, кэшированные в данный момент на клиенте, если таковые имеются, передают на сервер.

В ответ на прием запроса 1064 синхронизации драйвера, сервер 123 метаданных осуществляет ответ, посылая все обновления драйвера, которые применяют к клиентскому вычислительному устройству 110, еще не кэшированные на клиенте. Обновление драйвера посылают в ответе 1065 на клиентское вычислительное устройство 110, если его предварительные условия и условия применимости удовлетворены. Ответ 1065, передающий обновления драйвера, предпочтительно передает командный компонент каждого обновления. Обновления драйвера затем записывают в кэш обновлений клиентского вычислительного устройства.

После приема ответа 1065, содержащего обновления драйвера, подпрограмма 1050 синхронизации посылает запрос 1066 локализованных данных для каждого из принятых обновлений программного и аппаратного обеспечения. В качестве итога вышеприведенного компонент локализованных данных каждого обновления включает в себя общую информацию, описывающую обновление. Например, компонент локализованных данных может включать в себя информацию, описывающую функциональные возможности и эффект обновления. Компонент локализованных данных может также включать в себя текстовое описание процедур установки обновления. Кроме того, компонент локализованных данных может включать в себя любые другие данные или информацию, относящуюся к обновлению.

Таким образом, после приема запроса 1066 локализованных данных для каждого из принятых обновлений программного и аппаратного обеспечения сервер метаданных 123 осуществляет ответ, посылая все локализованные данные для всех принятых обновлений программного и аппаратного обеспечения, хранимых в кэше обновлений клиента. Как только приняты, локализованные данные могут быть обработаны программным приложением для того, чтобы определить, какое из обновлений должно быть установлено. В качестве альтернативы, принятые

локализованные данные могут быть визуально представлены пользователю, чтобы информировать пользователя обо всех обновлениях, которые являются доступными клиентскому вычислительному устройству 110. В одном варианте осуществления принятые локализованные данные могут быть отображены на web-странице. В

данном примере локализованные данные могут быть приняты клиентом для шестого и восьмого обновлений 921 и 931. Если локализованные данные хранятся в базовых обновлениях 911, 912, 913 и 915, локализованные данные для этих обновлений будут также приняты клиентом.

На Фиг.11 проиллюстрирован один пример web-страницы 1100, отображающей пример локализованных данных, связанных с обновлениями, которые являются доступными клиенту. Для иллюстративных целей web-страница 1100 содержит первое подробное описание 1105 обновления и второе подробное описание 1106 другого обновления. Также показано, что каждое обновление является соответственно связанным с механизмами (средствами) 1103 и 1104 выборки, предназначенными для приема пользовательской выборки обновлений. Также показано, что web-страница 1100 скомпонована с управляющей кнопкой 1101, чтобы дать возможность пользователю управлять передачей выборки обновлений на сервер, такой как сервер 123 метаданных или сервер 124 загрузки.

В одном аспекте настоящего изобретения клиент выполняет набор процессов, чтобы повысить значимость визуального представления web-страницы 1100. Например, клиентское вычислительное устройство 110 исследует локализованные данные для каждого обновления, чтобы определить, является ли конкретное обновление высокоприоритетным. Такой функциональной возможности можно содействовать, помещая текст в локализованные данные или в другой компонент конкретного обновления, который указывает, что конкретное обновление является высокоприоритетным обновлением или экстренным (аварийным) обновлением. Если клиентское вычислительное устройство 110 выявляет высокоприоритетное обновление или экстренное обновление, клиент отображает высокоприоритетное обновление в видимой части web-страницы 1100, например в верхней части страницы. Кроме того, клиент может формировать визуальный индикатор, например специальное текстовое сообщение 1120, указывающее, что обновление является высокоприоритетным обновлением.

Клиентское вычислительное устройство 110 может также исследовать локализованные данные каждого обновления, чтобы определить, требует ли конкретное обновление эксклюзивной установки, то есть обновление, содержащее файл установки, который не может быть устанавливаемым одновременно с файлом установки другого обновления. Такой функциональной возможности можно содействовать, помещая текст в локализованные данные или в другой компонент конкретного обновления, который указывает, что конкретное обновление требует эксклюзивной установки. Если клиентское вычислительное устройство 110 выявляет такое обновление, клиент отображает визуальный индикатор, например текстовое сообщения 1122, показанное на Фиг.11, с описанием обновлений, которые требуют эксклюзивной установки.

Возвращаясь к Фиг.7, программа 700 обновления программного обеспечения продолжается на этапе 708, на котором клиентское вычислительное устройство 110 принимает выборку обновлений. Как отмечено выше, в ответ на приведение в действие управляющей кнопки 1101, выборка одного или нескольких обновлений может быть получена сервером 123 метаданных или сервером 124 загрузки. Как

только выборка одного или нескольких обновлений принята, программа 700 обновления программного обеспечения продолжается на этапе 708, на котором обрабатывают выбранные обновления программного обеспечения.

В соответствии с еще одним аспектом настоящего изобретения служба 120 обновления программного обеспечения может обеспечивать способ выбора и передачи информации между службой обновления программного обеспечения и клиентским вычислительным устройством 110. Фиг.12А и 12В являются иллюстративными для подпрограммы 1200 обработки обновления программного обеспечения, осуществляемой клиентским вычислительным устройством 110 для того, чтобы извлекать и устанавливать запрошенное программное обеспечение в соответствии с настоящим изобретением. Как описано выше, подпрограмма 1200 обработки обновления программного обеспечения может быть осуществлена, как только была сформирована или принята выборка обновлений программного обеспечения. Со ссылкой на Фиг.12А, на этапе 1202 компонент 111 управления обновлением инициализирует агент 118 обновления. В иллюстративном варианте осуществления настоящего изобретения агентом 118 обновления является специальный программный компонент, предназначенный для определения, какая информация об обновлении программного обеспечения необходима для того, чтобы выполнить запрошенное обновление программного обеспечения, сформировать необходимую версию установочного компонента для агента обновления, сформировать обновленные файлы посредством объединения существующих файлов с дельта-корректировками, и/или инициировать установку обновленных файлов. В том случае, когда агент 118 обновления уже инициализирован, этап 1202 может быть опущен.

На этапе 1204 агент 118 обновления получает информацию об обновлении программного обеспечения от службы 120 обновления. В иллюстративном варианте осуществления настоящего изобретения информация об обновлении программного обеспечения, переданная службой 120 обновления, находится в форме пакета, такого как саморазворачивающийся файл, который включает в себя многообразие данных, которые могут быть использованы агентом обновления. В одном аспекте пакет может включать в себя перечень всех файлов, которые соответствуют конкретному обновлению программного обеспечения. Дополнительно, пакет может включать в себя копию по меньшей мере части из декларации хранилища корректировок, которая отображает конкретные версии файлов, подлежащих обновлению, на соответствующие дельта-корректировки обновлений программного обеспечения, хранимые в файле хранилища корректировок, используемом службой 120 обновления. Пакет может также включать в себя установочную информацию для каждого файла, подлежащего обновлению, которая может включать в себя идентификацию версии установочного компонента, необходимого для выполнения установки. Дополнительно, пакет также может включать в себя установочный компонент для агента 118 обновления или дельта-корректировку, чтобы обновить версию установочного компонента, уже хранимого на клиентском вычислительном устройстве 110. Кроме того, дополнительно пакет может включать в себя информацию верификации, чтобы позволить агенту обновления определять, было ли обновление программного обеспечения успешным. Например, информация верификации может включать в себя контрольные значения хэш-функции для обновленных файлов для сравнения. Агент 118 обновления может также проверять содержимое пакета.

На этапе 1206 принятия решения проводят проверку, чтобы определить, нуждается ли агент 118 обновления в том, чтобы обновить версию установочного компонента для осуществления обновления. Специалист в соответствующей области техники оценит, что передача полной копии установочного компонента в саморазворачивающемся файле может увеличивать объем данных, передаваемых службой 120 обновления для каждого обновления программного обеспечения. Соответственно, в иллюстративном варианте осуществления настоящего изобретения базовая версия установочного компонента может быть хранимой на клиентском вычислительном устройстве и обновляемой конкретно для требований текущего обновления программного обеспечения в виде дельта-корректировки установочного компонента. Соответственно, установочная информация в саморазворачивающемся файле информирует агента 118 обновления о том, должны ли или не должны включенные в него обновления установочного компонента быть объединены с базовой версией установочного компонента на клиентском вычислительном устройстве 110. Если обновление необходимо, то на этапе 1208 агент 118 обновления обновляет базовый установочный компонент, как будет объяснено более подробно ниже в отношении Фиг.13.

Как только агент обновления обновит установочный компонент или если установочный компонент не требует обновления, на этапе 1210 агент 118 обновления выполняет инвентаризацию файлов, установленных на клиентском вычислительном устройстве 110, и конкретной версии файла. В иллюстративном варианте осуществления настоящего изобретения агент 118 обновления может запросить файловую систему клиентского вычислительного устройства 110 обо всех файлах, идентифицированных в пакете в качестве соответствующих выбранному обновлению. В качестве альтернативы, если агент 118 обновления содержит недавно проведенную инвентаризацию, может быть использована кэшированная версия инвентаризационной информации. На этапе 1212 агент 118 обновления идентифицирует, какая информация об обновлении программного обеспечения необходима, чтобы выполнить требуемое обновление. В иллюстративном варианте осуществления настоящего изобретения декларация хранилища корректировок включает в себя отображение версий установленного файла на необходимую дельта-корректировку. Соответственно, если дельта-корректировка является доступной, агент 118 обновления использует отображение, чтобы идентифицировать конкретную дельта-корректировку и ячейку памяти смещения в пределах файла хранилища корректировок. В качестве альтернативы, если дельта-корректировка не является доступной или не может быть осуществлена, агент 118 обновления может идентифицировать полный файл для загрузки.

Со ссылкой теперь на Фиг.12В, на этапе 1214 агент обновления передает запрос идентифицированной информации обновления программного обеспечения. В иллюстративном варианте осуществления настоящего изобретения агент 118 обновления может передавать запрос конкретных дельта-корректировок посредством указания конкретной группы корректировок, запрашиваемых из файла хранилища корректировок на сервер 124 загрузки, относящийся к службе 120 обновления. Как описано выше, файл хранилища корректировок включает в себя большое количество применимых дельта-корректировок, в котором каждая дельта-корректировка идентифицирована ее местоположением внутри файла хранилища корректировок. Поскольку файл хранилища корректировок может быть довольно большим в некоторых осуществлениях, агент 118 обновления может использовать запрос,

который запрашивает из хранилища корректировок данные только из конкретных местоположений в файле хранилища корректировок, указываемых на основании декларации хранилища корректировок. В дополнительном варианте осуществления настоящего изобретения агент 118 обновления может запрашивать полную копию файла обновления и/или полную копию файла хранилища корректировок.

В альтернативном варианте осуществления настоящего изобретения другой сервер загрузки, который может не быть эксклюзивно связанным со службой 120 обновления, может обрабатывать запрос агента 118 обновления. В этом варианте осуществления запрошенный файл хранилища корректировок может быть передан полностью или частично на любое количество дополнительных серверов загрузки в сети.

Дополнительные серверы загрузки могут быть частью частной сети, используемой для обновления клиентов в частной сети. Кроме того, дополнительный сервер загрузки может быть частью общедоступной сети. В среде частной сети серверы загрузки могут получать полную копию файла хранилища корректировок для обработки запросов клиента. В качестве альтернативы, серверы загрузки могут также кэшировать части файла хранилища корректировок в ходе обработки предыдущих запросов данных от других клиентов и использовать данные кэша, чтобы выполнить загрузку.

Соответственно, дополнительные серверы загрузки могут сократить нагрузку передачи данных на сервер 124 загрузки службы 120 обновления.

На этапе 1216 агент 118 обновления принимает запрошенную информацию об обновлении. В иллюстративном варианте осуществления настоящего изобретения запрошенная информация об обновлении может быть передана с применением двух подходов. В первом подходе, названном как ручное обновление, запрос обновления посылают на службу 120 обновления с запросом на непосредственную доставку HTTP данных в ответ. В этом подходе служба 120 обновления может использовать всю полную полосу частот, доступную для передачи запрошенных данных на агент 118 обновления. Во втором подходе, названном как автоматическое обновление, запрос обновления посылают службе 120 обновления с запросом на косвенную доставку HTTP данных в ответ. В этом ответе служба 120 обновления передает запрошенные данные как фоновый процесс. Фоновый процесс может быть осуществлен таким образом, чтобы использовать минимальный объем доступной полосы частот.

Дополнительно, фоновый процесс может быть прерван в течение процесса загрузки и повторно запущен в седующий доступный момент времени. Описание системы и способа передачи запрошенных данных посредством фонового процесса описаны в совместно переданной и рассматриваемой одновременно заявке на патент США за номером №09/505,735, озаглавленной «System and Method for Transferring Data Over a Network», поданной 16 февраля 2000, которая тем самым включена в настоящий документ путем ссылки. Специалист в соответствующей области техники оценит, что (высоко)приоритетная или низкоприоритетная (фоновая) доставка данных необязательно отражает приоритет выбранного обновления программного обеспечения, а предпочтительнее то, как распределена полоса частот, чтобы получать информацию об обновлении.

Как только запрошенная информация принята от службы обновления, на этапе 1218 агент 118 обновления объединяет дельта-корректировку с соответствующими установленными файлами. В иллюстративном варианте осуществления настоящего изобретения агент 118 обновления может кэшировать исходную версию установленного файла, чтобы гарантировать, что выбранный файл не изменяется в течение процесса загрузки и объединения. Дополнительно,

кэшированная исходная версия установленного файла может быть использована, чтобы удалять установленное выбранное обновление.

На этапе 1220 принятия решения проводится проверка, чтобы определить, является ли обновленный файл действительным. В иллюстративном варианте осуществления настоящего изобретения агент 118 обновления может использовать алгоритм хэширования, чтобы сравнивать контрольное хэш-значение, полученное из пакета информации об обновлении и соответствующее действительному файлу обновления, с хэш-значением из текущего модифицированного файла. Если хэш-значения не совпадают, то текущий модифицированный файл не является действительным. Специалист в соответствующей области техники оценит, что любой из набора альтернативных алгоритмов проверки достоверности также может быть использован. Если обновленный файл не является действительным, подпрограмма 1200 осуществляет возврат на этап 1214, на котором агент обновления может вновь запрашивать информацию об обновлении. В качестве альтернативы, если агент 118 обновления неудачно пытался сформировать файл обновления несколько раз, то агент обновления может осуществить одну из нескольких процедур восстановления. В одном варианте осуществления настоящего изобретения агент 118 обновления может запрашивать полную копию обновленного файла, хранимого в файле хранилища корректировок и идентифицированного на основании декларации хранилища корректировок, от службы 120 обновления. В другом варианте осуществления настоящего изобретения агент 118 обновления может запрашивать от службы 120 обновления копию обновленного файла в отдельном файле. В еще одном варианте осуществления настоящего изобретения подпрограмма 1200 может в противном случае терпеть неудачу.

Если выбранный файл является действительным, на этапе 1222 принятия решения проводят проверку, чтобы определить, необходимы ли какие-либо дополнительные загрузки. В иллюстративном варианте осуществления настоящего изобретения подпрограмма 1200 входит в итерационный цикл, который постоянно проверяет наличие дополнительных загрузок после выполнения предварительно выбранной загрузки. Если состояние файла изменяется в течение загрузки, агент 118 обновления будет продолжать запрашивать о дополнительных загрузках для нового изменения состояния. Если дополнительные загрузки необходимы, то на этапе 1224 агент 118 обновления выполняет другую инвентаризацию и идентифицирует все применимые дельта-корректировки. Подпрограмма 1200 затем возвращается на этап 1214.

Как только все запрошенные загрузки обновления выполнены, на этапе 1226 принятия решения проводится проверка, чтобы определить, изменилось ли состояние клиентского устройства. В иллюстративном варианте осуществления настоящего изобретения может истечь время между загрузкой и объединением информации об обновлении и фактической установкой обновленного файла. Соответственно, до установки обновленного файла агент обновления определяет, изменилось ли состояние клиентского вычислительного устройства. Если состояние изменилось, обновление файла может не быть действительным, и обновление терпит неудачу на этапе 1228. В качестве альтернативы, если изменение состояния не произошло, агент 118 обновления устанавливает обновленный файл на этапе 1230 и подпрограмма 1200 возвращается на этап 1232.

Со ссылкой теперь на Фиг.13 описана подпрограмма 1300, осуществляемая клиентским вычислительным устройством 110 для обновления базового установочного компонента, соответствующая этапу 1208 (Фиг.12А). На этапе 1302

принятия решения проводят проверку, чтобы определить, включен ли новый базовый установочный компонент в саморазворачивающийся файл, передаваемый агенту обновления 118 от службы 120 обновления. В иллюстративном варианте осуществления настоящего изобретения, если дельта-корректировки, необходимые для обновления базовой программы установки, сопоставимы по размеру с передачей обновленного установочного компонента, то будет передан новый базовый установочный компонент. Если обновленный установочный компонент включен в состав, то на этапе 1304 агент обновления устанавливает обновленный базовый установочный компонент в качестве нового установочного компонента.

Дополнительно, новый обновленный установочный компонент может быть сохранен в запоминающем устройстве клиентского вычислительного устройства 110, чтобы использовать в качестве базовой программы установки для дополнительных обновлений. На этапе 1306 подпрограмма осуществляет возврат.

Если обновляемый базовый установочный компонент не включен в саморазворачивающийся файл, то на этапе 1308 агент 118 обновления получает дельта-корректировку базового установочного компонента из саморазворачивающегося файла. В иллюстративном варианте осуществления настоящего изобретения дельта-корректировка базового установочного компонента соответствует программному коду, который может быть объединен с базовым установочным компонентом, чтобы сформировать обновленный базовый установочный компонент. Соответственно, на этапе 1310 обновленный агент объединяет дельта-корректировку базового установочного компонента с базовым установочным компонентом. На этапе 1312 агент 118 обновления затем обозначает обновленный базовый установочный компонент в качестве текущего установочного компонента. В иллюстративном варианте осуществления настоящего изобретения обновленный установочный компонент не будет сохранен после того, как установка выполнена. В соответствии с данным вариантом осуществления агент 118 обновления поддерживает только ограниченное количество базовых установочных компонентов в запоминающем устройстве клиентского вычислительного устройства 110.

Соответственно, агент обновления формирует временный обновленный установочный компонент при каждой установке. Поскольку каждое клиентское вычислительное устройство 110 может передавать только ограниченное количество базовых установочных компонентов, службе 120 обновления необходимо передавать только единственную дельта-корректировку базового установочного компонента для каждого клиентского вычислительного устройства. На этапе 1314 подпрограмма 1300 осуществляет возврат.

Несмотря на то, что был проиллюстрирован и описан предпочтительный вариант осуществления изобретения, очевидно, что в нем могут быть сделаны различные изменения без выхода за пределы существа и объема изобретения. Например, хотя иллюстративные примеры, описанные при этом, применяют к обновлениям программного обеспечения, объем настоящего изобретения включает в себя другие использования вне распространения и передачи информации, относящейся к обновлениям программного обеспечения. Соответственно, если объект изобретения не является явно исключенным в данном раскрытии, то должно быть понятно, что объем настоящего изобретения применяется к распространению и передаче данных любого типа, отличных от обновления программного обеспечения, или в дополнение к ним.

Варианты осуществления изобретения, в которых заявлены исключительное право собственности или право использования, определены формулой изобретения в

качестве нижеследующих.

Формула изобретения

1. Способ обмена информацией между клиентским вычислительным устройством и серверным вычислительным устройством, содержащий этапы:

а) выдают посредством клиентского вычислительного устройства запрос синхронизации, причем запрос синхронизации включает в себя идентификатор установленного обновления программного обеспечения, если клиентское вычислительное устройство хранит установленное обновление программного обеспечения;

б) принимают посредством клиентского вычислительного устройства командные компоненты множества обновлений программного обеспечения в ответ на запрос синхронизации, причем каждый командный компонент содержит по меньшей мере одно правило применимости, при этом упомянутое по меньшей мере одно правило применимости определяет по меньшей мере одно условие, требуемое клиентским вычислительным устройством до установки выбранного обновления программного обеспечения, ассоциированного с упомянутым командным компонентом;

в) сохраняют посредством клиентского вычислительного устройства упомянутый командный компонент каждого обновления программного обеспечения в клиентском вычислительном устройстве;

г) запрашивают посредством клиентского вычислительного устройства компонент локализованных данных по меньшей мере первого из обновлений программного обеспечения на основании, по меньшей мере частично, определения клиентским вычислительным устройством, что упомянутое по меньшей мере одно условие соответствующего правила применимости удовлетворяется, и

е) принимают в клиентском вычислительном устройстве компонент локализованных данных упомянутого первого обновления программного обеспечения, причем каждый компонент локализованных данных включает в себя дополнительную информацию, описывающую соответствующее обновление программного обеспечения.

2. Способ по п.1, дополнительно содержащий повторение посредством клиентского вычислительного устройства этапов (а)-(е), если командный компонент первого обновления программного обеспечения обеспечивает индикацию того, что дополнительные обновления программного обеспечения являются доступными для обработки клиентским вычислительным устройством.

3. Способ по п.1, в котором:

клиентское вычислительное устройство вызывает приложение для генерации отображения компонента локализованных данных первого обновления программного обеспечения на устройстве отображения после приема компонента локализованных данных.

4. Способ по п.1, дополнительно содержащий:

принимают посредством клиентского вычислительного устройства команду для запуска передачи файла данных, связанного с первым обновлением программного обеспечения; и

выдают посредством клиентского вычислительного устройства команду на передачу файла данных, связанного с первым обновлением программного обеспечения; и

принимают посредством клиентского вычислительного устройства загрузку файла

данных.

5. Способ по п.1, в котором сохранение командного компонента каждого обновления программного обеспечения в клиентском вычислительном устройстве содержит сохранение командного компонента каждого обновления программного обновления в качестве установленного обновления программного обновления, если клиентское вычислительное устройство содержит по меньшей мере один компонент, который удовлетворяет упомянутому по меньшей мере одному условию по меньшей мере одного правила применимости, сохраненного в обновлении программного обеспечения, и в котором способ дополнительно содержит этапы:

включение посредством клиентского вычислительного устройства идентификатора установленного обновления программного обеспечения в запрос синхронизации, если клиентское вычислительное устройство хранит это установленное обновление программного обеспечения.

6. Способ по п.1, в котором сохранение командного компонента каждого обновления программного обновления в клиентском вычислительном устройстве содержит этапы:

сохраняют посредством клиентского вычислительного устройства командные компоненты обновлений программного обеспечения в клиентском вычислительном устройстве в качестве неуспешных обновлений программного обеспечения, если клиентское вычислительное устройство определяет, что по меньшей мере одно условие упомянутого по меньшей мере одного правила применимости, хранимого в каждом соответствующем обновлении программного обеспечения, не выполнено.

7. Способ по п.1, в котором дополнительно:

включают посредством клиентского вычислительного устройства идентификатор любых неуспешных обновлений программного обеспечения в запрос синхронизации, причем прием командных компонентов обновлений программного обеспечения в ответ на этот запрос синхронизации содержит прием командных компонентов только обновлений программного обеспечения, которые являются не относящимися к неуспешным обновлениям программного обеспечения.

8. Способ обмена информацией между клиентским обрабатывающим устройством и серверным обрабатывающим устройством в системе распределенной обработки данных, содержащий этапы:

а) выдают посредством клиентского обрабатывающего устройства запрос авторизации, причем запрос авторизации содержит идентификатор клиентского модуля авторизации, связанного с целевой группой;

б) принимают посредством клиентского обрабатывающего устройства серверные куки-данные, идентифицирующие упомянутую целевую группу;

с) выдают посредством клиентского обрабатывающего устройства запрос обновления на обновления программного обеспечения, причем запрос обновления содержит серверные куки-данные;

д) принимают посредством клиентского обрабатывающего устройства командный компонент для по меньшей мере одного обновления программного обеспечения, связанного с целевой группой, причем каждый командный компонент содержит по меньшей мере одно правило применимости, при этом упомянутое по меньшей мере одно правило применимости определяет по меньшей мере одно условие, требуемое клиентским обрабатывающим устройством до установки выбранного обновления программного обеспечения, ассоциированного с упомянутым командным компонентом;

е) запрашивают посредством клиентского обрабатывающего устройства компонент локализованных данных для по меньшей мере первого из обновлений программного обеспечения на основании, по меньшей мере частично, определения посредством клиентского обрабатывающего устройства, что упомянутое по меньшей мере одно условие соответствующего правила применимости удовлетворено; и

ф) принимают посредством клиентского обрабатывающего устройства компонент локализованных данных упомянутого первого обновления программного обеспечения, причем упомянутый компонент локализованных данных включает в себя дополнительную информацию, описывающую это первое обновление программного обеспечения.

9. Способ по п.8, в котором запрос авторизации включает в себя данные предварительно сохраненных серверных куки-данных, и при этом серверные куки-данные, выдаваемые серверным обрабатывающим устройством, включают в себя данные предварительно сохраненных серверных куки-данных.

10. Способ по п.8, в котором определяют, что клиентское обрабатывающее устройство является связанным с целевой группой, если серверный модуль авторизации в обмене информацией с серверным обрабатывающим устройством идентифицирует ту же самую целевую группу, связанную с клиентским модулем авторизации.

11. Способ по п.8, в котором серверные куки-данные содержат срок действия, причем прием командного компонента каждого обновления программного обеспечения включает в себя прием командного компонента только если срок действия серверных куки-данных не истек.

12. Способ по п.11, в котором серверные куки-данные хранят срок действия в формате открытого текста и в зашифрованном формате, и при этом серверные куки-данные содержат зашифрованные данные, идентифицирующие по меньшей мере одну целевую группу, причем способ дополнительно содержит этапы:

получают посредством клиентского обрабатывающего устройства срок действия из серверных куки-данных, причем клиентское обрабатывающее устройство получает срок действия из срока действия, хранимого в формате открытого текста; и

выдают посредством клиентского обрабатывающего устройства запрос на по меньшей мере одно обновление программного обеспечения, причем запрос включает в себя серверные куки-данные.

13. Способ по п.8, в котором, если командный компонент одного из обновлений программного обеспечения указывает, что доступны дополнительные обновления программного обеспечения, то способ дополнительно содержит этап:

обрабатывают посредством клиентского обрабатывающего устройства компонент локализованных данных обновления программного обеспечения для выбора по меньшей мере одного из дополнительных обновлений.

14. Система обмена и управления обновлениями программного обеспечения, содержащая:

(а) первую базу данных, хранящую множество обновлений программного обеспечения, причем каждое индивидуальное обновление программного обеспечения, хранимое в базе данных, содержит:

(i) поле командных данных, содержащее правило применимости, задающее по меньшей мере одно условие, подлежащее удовлетворению до установки индивидуального обновления программного обеспечения, при этом поле командных данных также содержит набор предварительных условий, которые идентифицируют

любые обновления программного обеспечения, которые требуются для установки этого индивидуального обновления программного обеспечения;

(ii) поле локализованных данных, содержащее текстовые данные, описывающие дополнительную информацию, относящуюся к индивидуальному обновлению программного обеспечения; и

(iii) компонент данных, который идентифицирует по меньшей мере один файл потока данных, связанный с упомянутым индивидуальным обновлением программного обеспечения;

(b) серверное вычислительное устройство авторизации для авторизации доступа к упомянутым обновлениям программного обеспечения, хранимым в первой базе данных, причем серверное вычислительное устройство авторизации выдает серверные куки-данные к клиентскому вычислительному устройству, чтобы разрешить этому клиентскому вычислительному устройству доступ к индивидуальному обновлению программного обеспечения, если клиентское вычислительное устройство содержит модуль авторизации, указывающий, что это клиентское вычислительное устройство является связанным с целевой группой клиентских вычислительных устройств, при этом серверные куки-данные содержат идентификатор для этой целевой группы, и при этом серверные куки-данные передаются на клиентское вычислительное устройство для сохранения серверных куки-данных;

(c) серверное вычислительное устройство метаданных для выдачи поля командных данных и поля локализованных данных по меньшей мере одного из обновлений программного обеспечения на клиентское вычислительное устройство, причем поле командных данных и поле локализованных данных каждого индивидуального обновления программного обеспечения передаются на клиентское вычислительное устройство, если индивидуальное обновление программного обеспечения связано с упомянутой целевой группой, при этом серверное вычислительное устройство метаданных коммуникативно соединено с упомянутой первой базой данных, и при этом серверное вычислительное устройство метаданных выполнено с возможностью:

(i) получать запрос синхронизации, причем запрос синхронизации включает в себя идентификатор установленного обновления программного обеспечения, если клиентское вычислительное устройство хранит установленное обновление программного обеспечения, при этом запрос синхронизации включает в себя серверные куки-данные;

(ii) выбирать дополнительное обновление программного обеспечения из обновлений программного обеспечения, хранимых в первой базе данных, для передачи на клиентское вычислительное устройство, если запрос синхронизации содержит идентификатор по меньшей мере одного установленного обновления программного обеспечения, причем выбор дополнительного обновления программного обеспечения зависит от выполнения предварительного условия, заданного в дополнительном обновлении программного обеспечения, при этом упомянутое предварительное условие требует, чтобы запрос синхронизации включал в себя идентификатор для упомянутого по меньшей мере одного установленного обновления программного обеспечения, при этом дополнительное обновление программного обеспечения является связанным с целевой группой, идентифицированной в серверных куки-данных;

(iii) выбирать обновление программного обеспечения первого уровня из обновлений программного обеспечения, хранимых в первой базе данных, для передачи на клиентское вычислительное устройство, если запрос синхронизации не

содержит идентификатор по меньшей мере одного установленного обновления программного обеспечения, при этом обновление программного обеспечения первого уровня не включает в себя предварительное условие;

(iv) передавать упомянутое поле командных данных выбранного обновления программного обеспечения на клиентское вычислительное устройство для сохранения;

(v) получать запрос на упомянутое поле локализованных данных, связанное с упомянутым выбранным обновлением программного обеспечения;

(vi) передавать поле локализованных данных, связанное с выбранным обновлением программного обеспечения, на клиентское вычислительное устройство, причем это поле локализованных данных позволяет клиентскому вычислительному устройству формировать запрос загрузки для приема упомянутого компонента данных выбранного обновления программного обеспечения; и

(d) серверное вычислительное устройство загрузки, коммуникативно соединенное с первой базой данных, и выполненное с возможностью:

(i) получать запрос загрузки от клиентского вычислительного устройства, причем запрос загрузки идентифицирует компонент данных выбранного обновления программного обеспечения; и

(ii) передавать компонент данных упомянутого выбранного обновления программного обеспечения на клиентское вычислительное устройство.

15. Система по п.14, в которой серверные куки-данные содержат идентификатор целевой группы, если серверный модуль авторизации идентифицирует целевую группу.

16. Система по п.14, в которой серверные куки-данные содержат срок действия, и при этом серверное вычислительное устройство метаданных выполнено с возможностью:

определить, истек ли срок действия серверных куки-данных; и

ограничивать передачу выбранного обновления программного обеспечения, если определено, что срок действия серверных куки-данных истек.

17. Система по п.14, в которой серверное вычислительное устройство метаданных выполнено с возможностью:

получать идентификатор неуспешного обновления программного обеспечения в запросе синхронизации, если клиентское вычислительное устройство сохраняет неуспешное обновление программного обеспечения; и

ограничивать передачу выбранных обновлений программного обеспечения, относящихся к упомянутому неуспешному обновлению программного обеспечения, если определено, что запрос синхронизации включает в себя идентификатор неуспешного обновления программного обеспечения.

18. Система по п.14, в которой правило применимости включает в себя нулевое значение.

19. Система по п.14, в которой набор предварительных условий включает в себя нулевое значение.

20. Система передачи обновлений программного обеспечения на клиентское вычислительное устройство, содержащая:

первый компонент, хранящий множество серверных модулей авторизации, причем индивидуальный серверный модуль авторизации из множества серверных модулей авторизации идентифицирует целевую группу клиентских вычислительных устройств; серверный компьютер, коммуникативно соединенный с упомянутым первым компонентом, причем серверный компьютер выполнен с возможностью:

получать запрос авторизации от клиентского вычислительного устройства, причем

запрос авторизации содержит идентификатор клиентского модуля авторизации, хранимого на клиентском вычислительном устройстве;

определять, является ли клиентское вычислительное устройство связанным с упомянутой целевой группой, причем определяют, что клиентское вычислительное устройство является связанным с этой целевой группой, если клиентский модуль авторизации указывает, что клиентское вычислительное устройство является связанным с целевой группой;

передавать серверные куки-данные от службы обновления программного обеспечения на клиентское вычислительное устройство, если определено, что клиентское вычислительное устройство является связанным с упомянутой целевой группой, причем серверные куки-данные идентифицируют упомянутую целевую группу;

получать запрос на по меньшей мере одно обновление программного обеспечения, хранимое службой обновления программного обеспечения, причем запрос содержит серверные куки-данные;

определять, является ли обновление программного обеспечения связанным с целевой группой, идентифицированной в серверных куки-данных, в ответ на получение запроса на по меньшей мере одно обновление программного обеспечения;

выдавать командный компонент обновлений программного обеспечения, если определено, что обновление программного обеспечения является связанным с целевой группой, идентифицированной в серверных куки-данных, причем каждый командный компонент содержит по меньшей мере одно правило применимости, определяющее по меньшей мере одно условие, требуемое клиентским вычислительным устройством до установки обновления программного обеспечения, связанного с командным компонентом,

принимать запрос на компонент локализованных данных упомянутого первого обновления программного обеспечения,

выдавать компонент локализованных данных упомянутого обновления программного обеспечения, причем компонент локализованных данных описывает упомянутое обновление программного обеспечения, и

передавать обновление программного обеспечения от службы обновления программного обеспечения на клиентское вычислительное устройство, если клиентское вычислительное устройство запрашивает упомянутое обновление программного обеспечения после приема компонента локализованных данных.

21. Способ передачи информации между клиентским вычислительным устройством и серверным вычислительным устройством, содержащий этапы:

(а) принимают серверным вычислительным устройством запрос синхронизации, при этом запрос синхронизации включает в себя идентификатор установленного обновления программного обеспечения, если клиентское вычислительное устройство хранит это установленное обновление программного обеспечения;

(б) выбирают серверным вычислительным устройством подходящие обновления программного обеспечения для клиентского вычислительного устройства на основе запроса синхронизации,

(с) выдают серверным вычислительным устройством командный компонент каждого из подходящих обновлений программного обеспечения в ответ на запрос синхронизации, причем каждый командный компонент содержит по меньшей мере одно правило применимости, причем упомянутое по меньшей мере одно правило применимости задает по меньшей мере одно условие, требуемое клиентским

вычислительным устройством до установки выбранного обновления программного обеспечения, связанного с упомянутым командным компонентом;

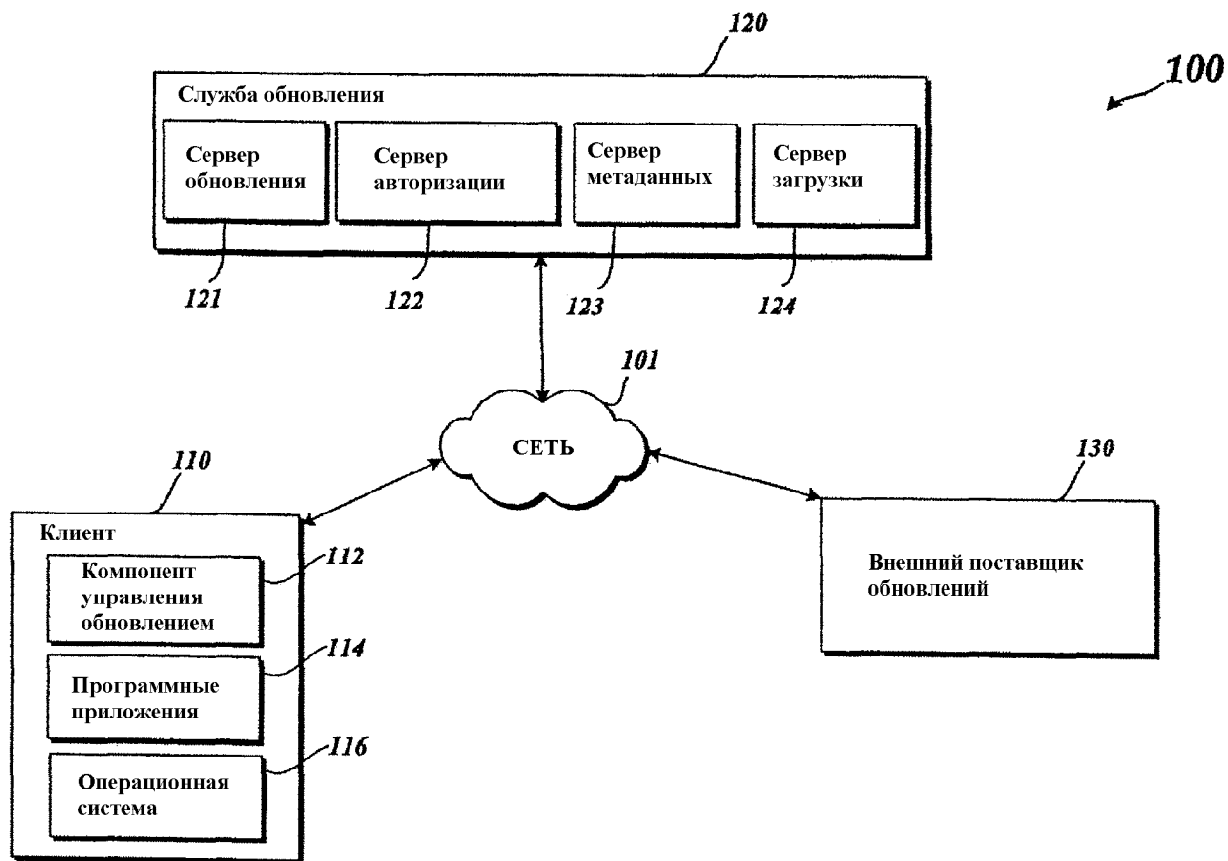
(d) принимают в серверном вычислительном устройстве запрос на компонент локализованных данных по меньшей мере первого обновления программного обеспечения из подходящих обновлений программного обеспечения на основании, по меньшей мере частично, определения клиентским вычислительным устройством, что это по меньшей мере одно условие соответствующего правила применимости удовлетворяется, и

(e) выдают серверным вычислительным устройством компонент локализованных данных первого обновления программного обеспечения, причем каждый компонент локализованных данных включает в себя дополнительную информацию, описывающую соответствующее подходящее обновление программного обеспечения.

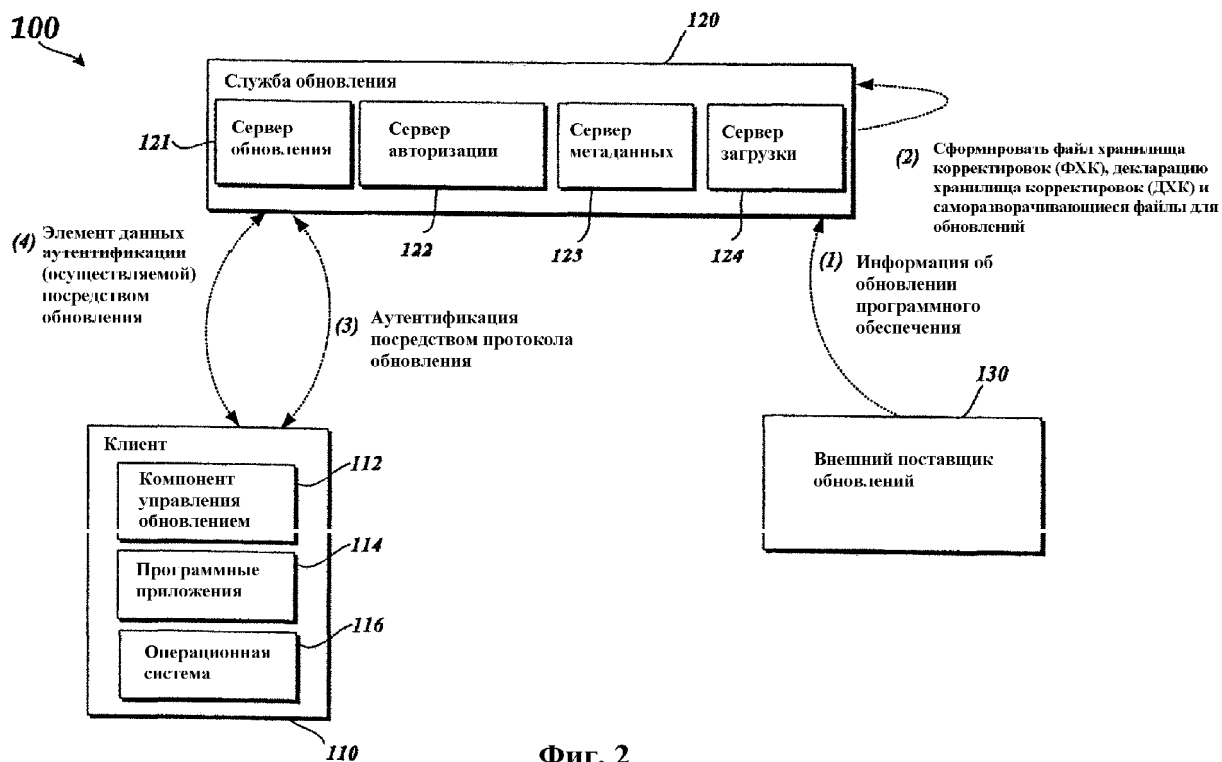
22. Способ по п.21, в котором выбор серверным вычислительным устройством по меньшей мере одного обновления программного обеспечения содержит этапы:

выбор серверным вычислительным устройством дополнительного обновления программного обеспечения, если запрос синхронизации включает в себя идентификатор по меньшей мере одного обновления программного обеспечения, причем выбор дополнительного обновления программного обеспечения зависит от удовлетворения предварительного условия, заданного в дополнительном обновлении программного обеспечения, причем упомянутое предварительное условие требует, чтобы запрос синхронизации включал в себя идентификатор по меньшей мере одного конкретного обновления программного обеспечения, и

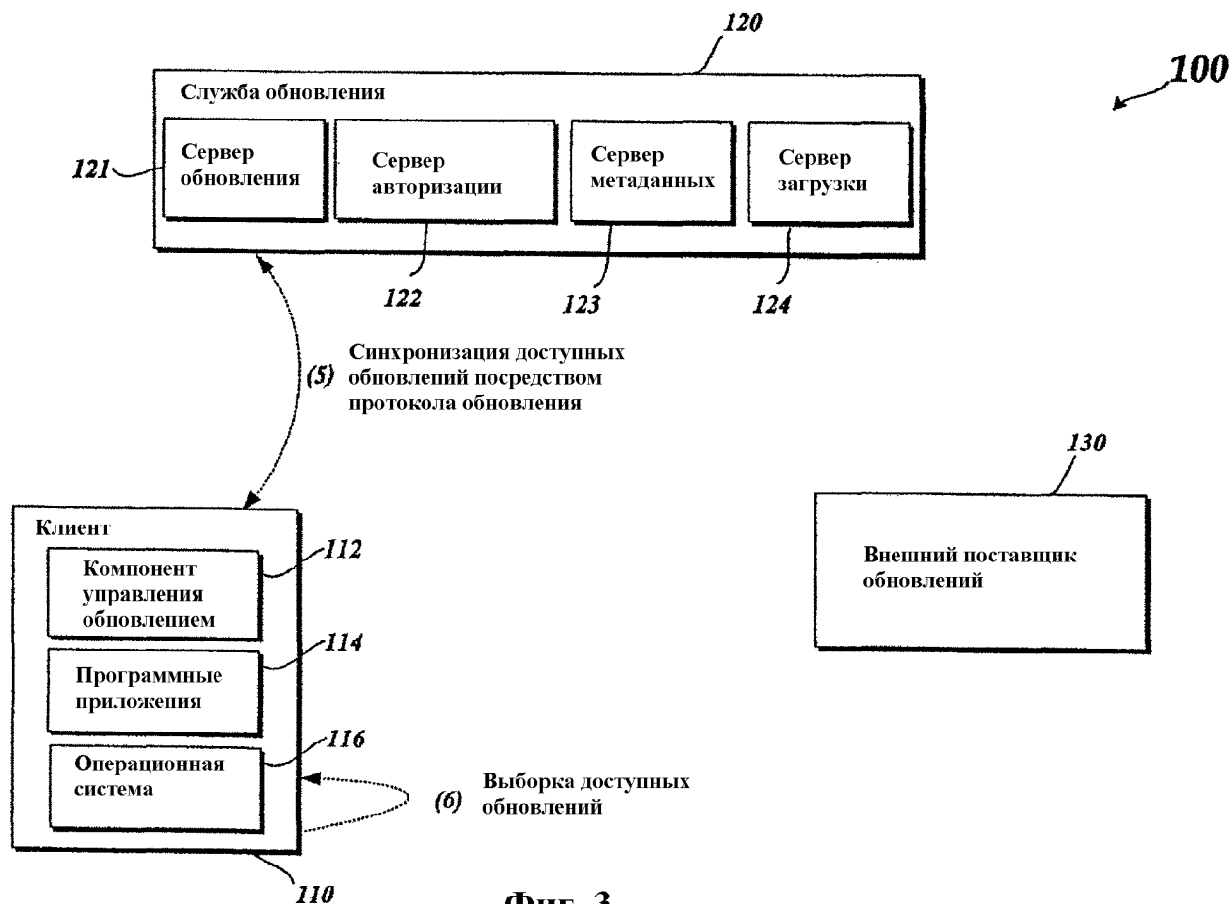
выбор серверным вычислительным устройством обновления программного обеспечения первого уровня, имеющее нулевое значение в качестве предварительного условия, если запрос синхронизации не включает в себя идентификатор по меньшей мере одного обновления программного обеспечения.



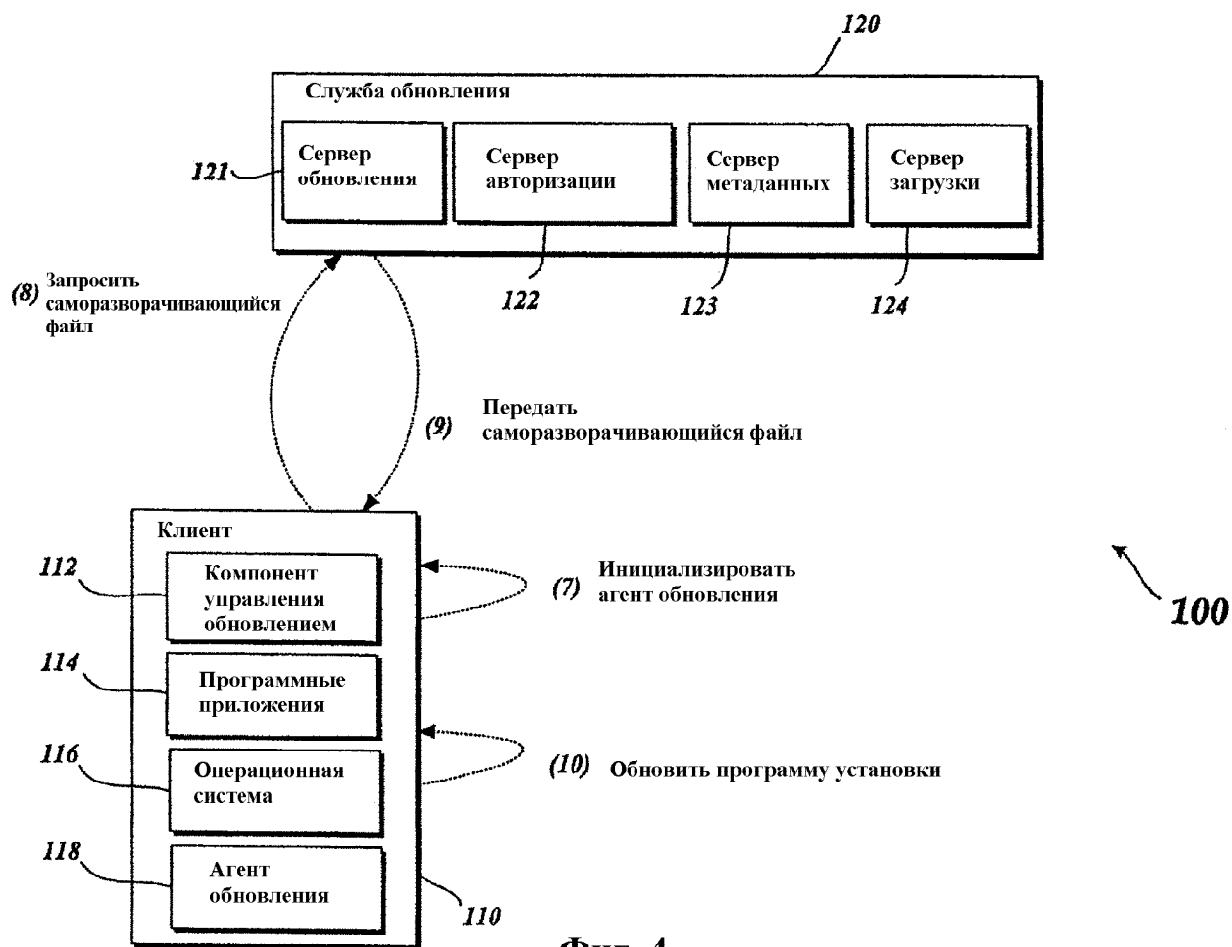
Фиг. 1



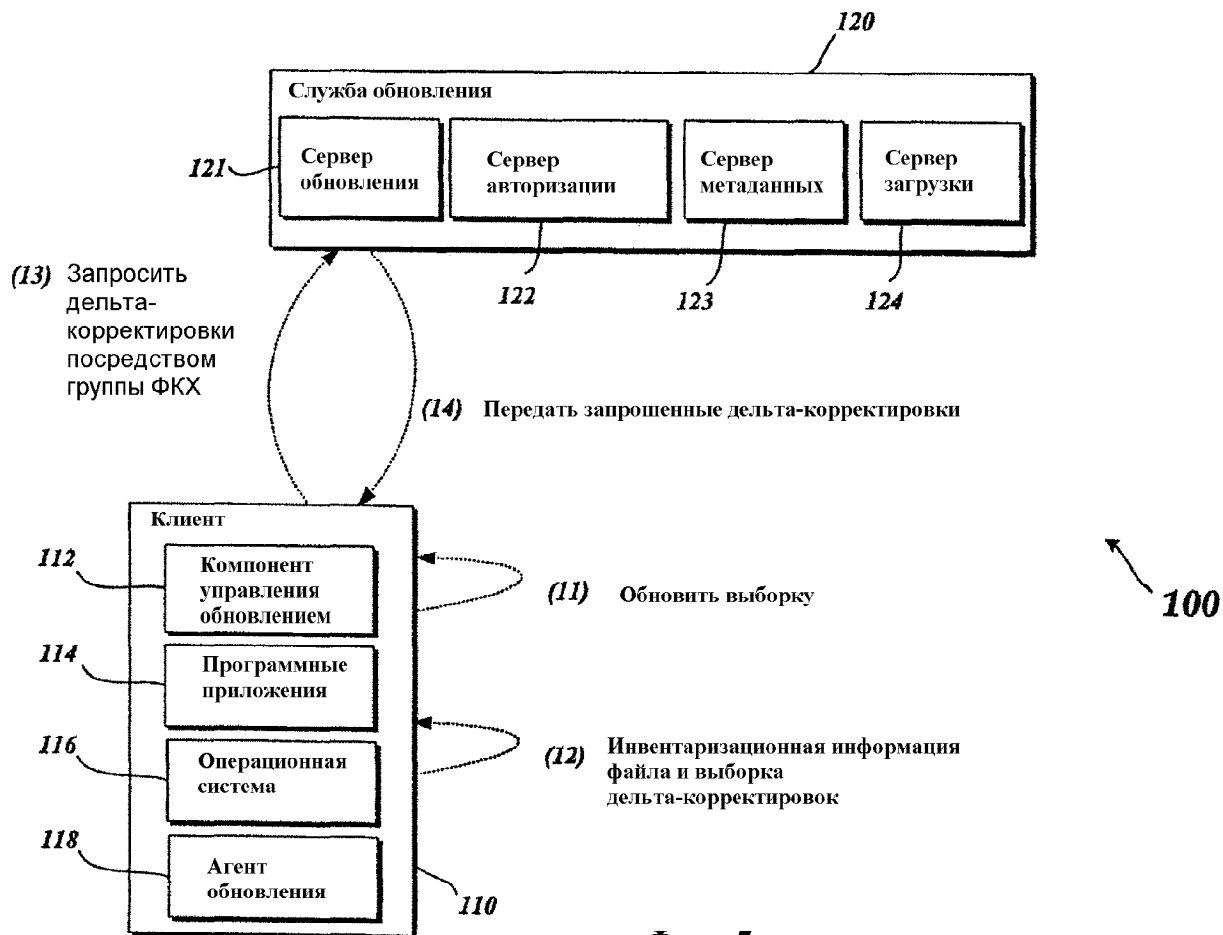
Фиг. 2



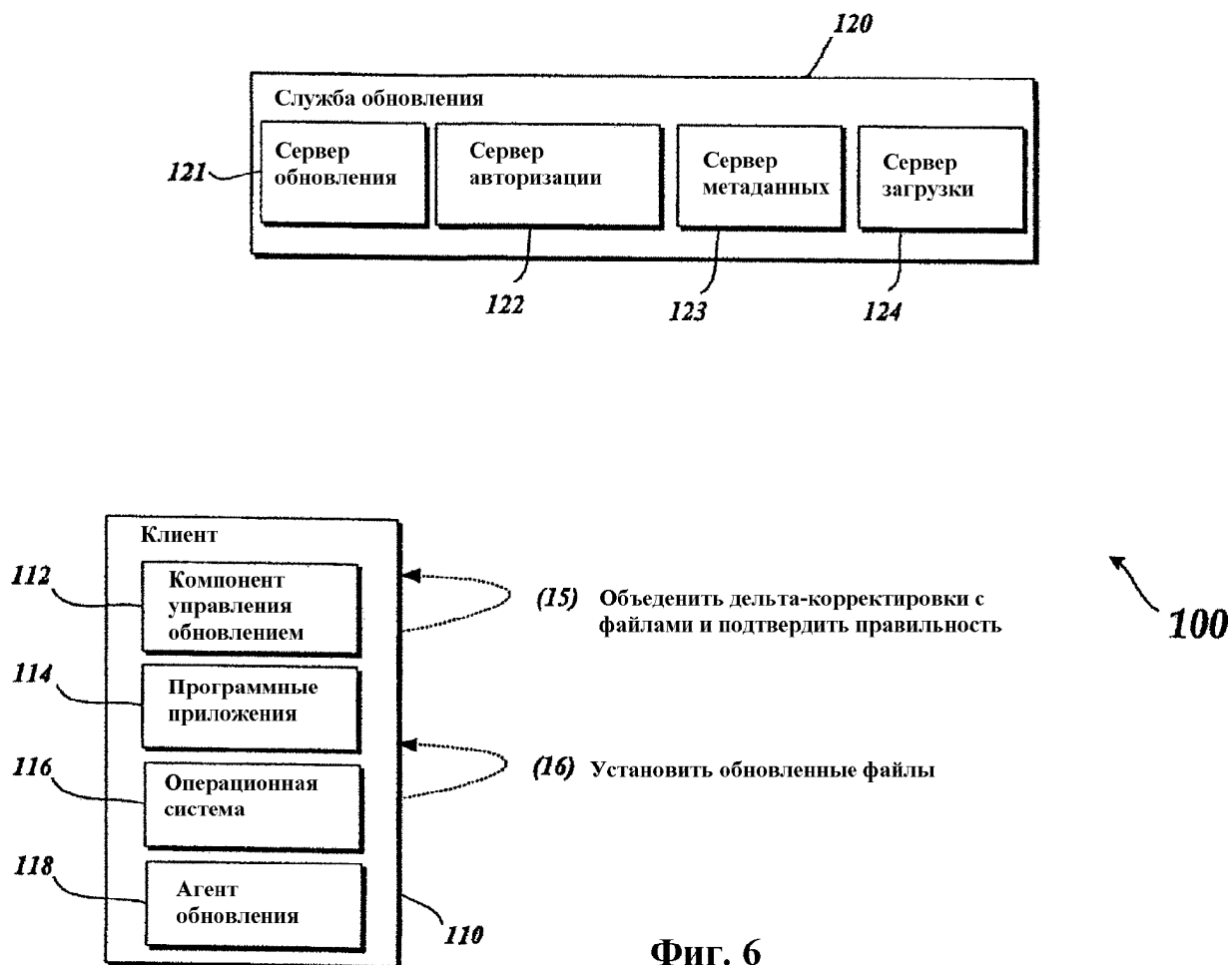
Фиг. 3



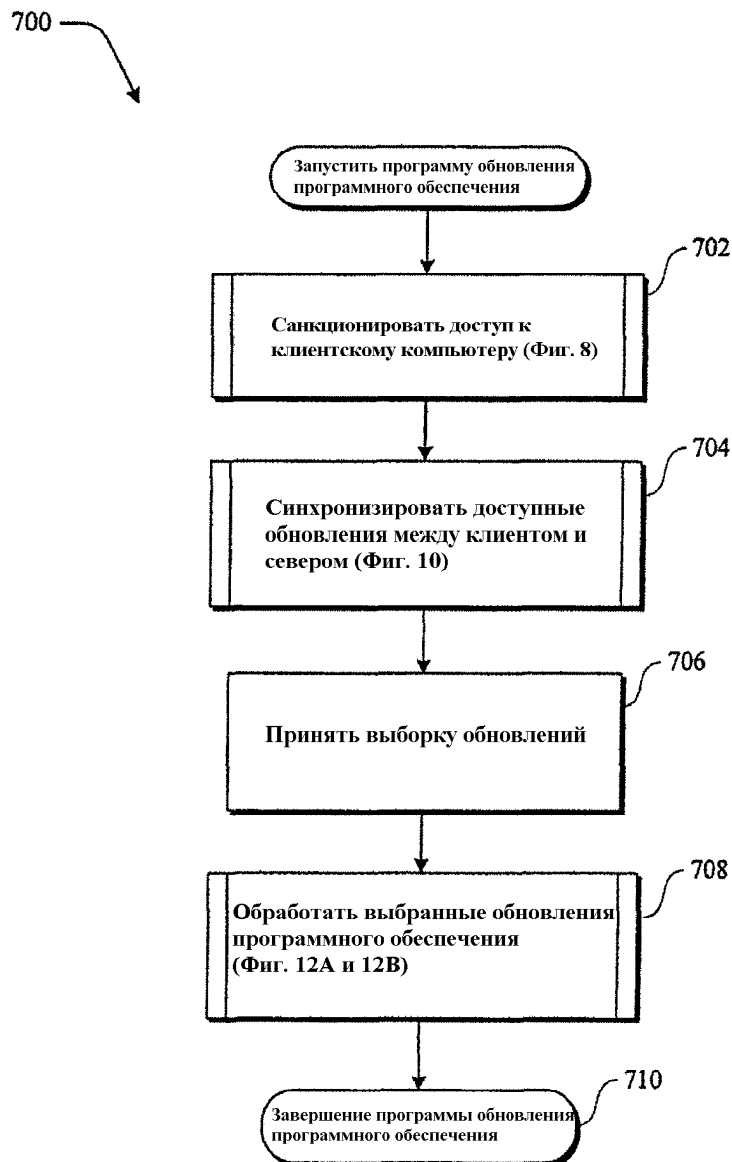
Фиг. 4



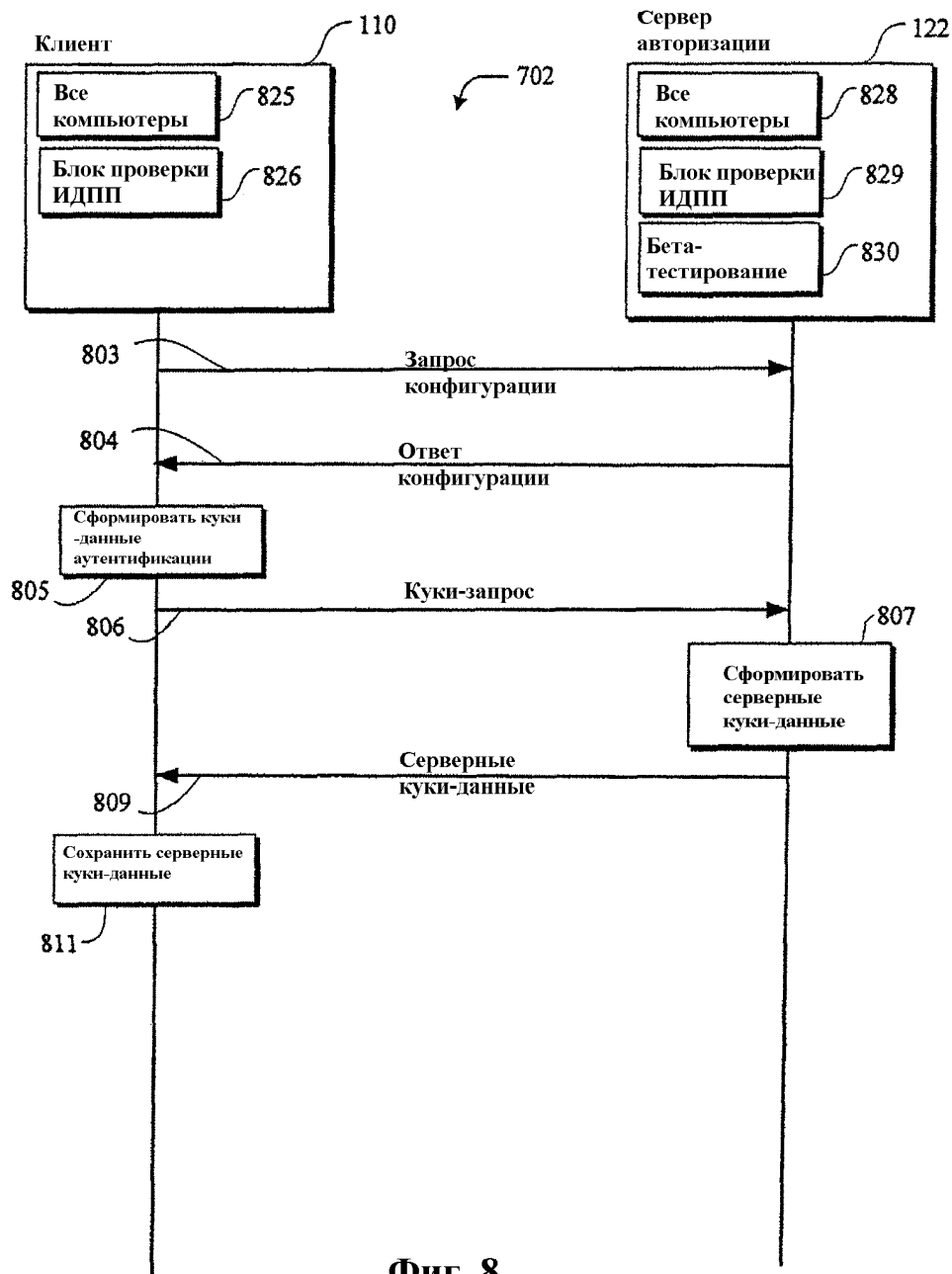
Фиг. 5



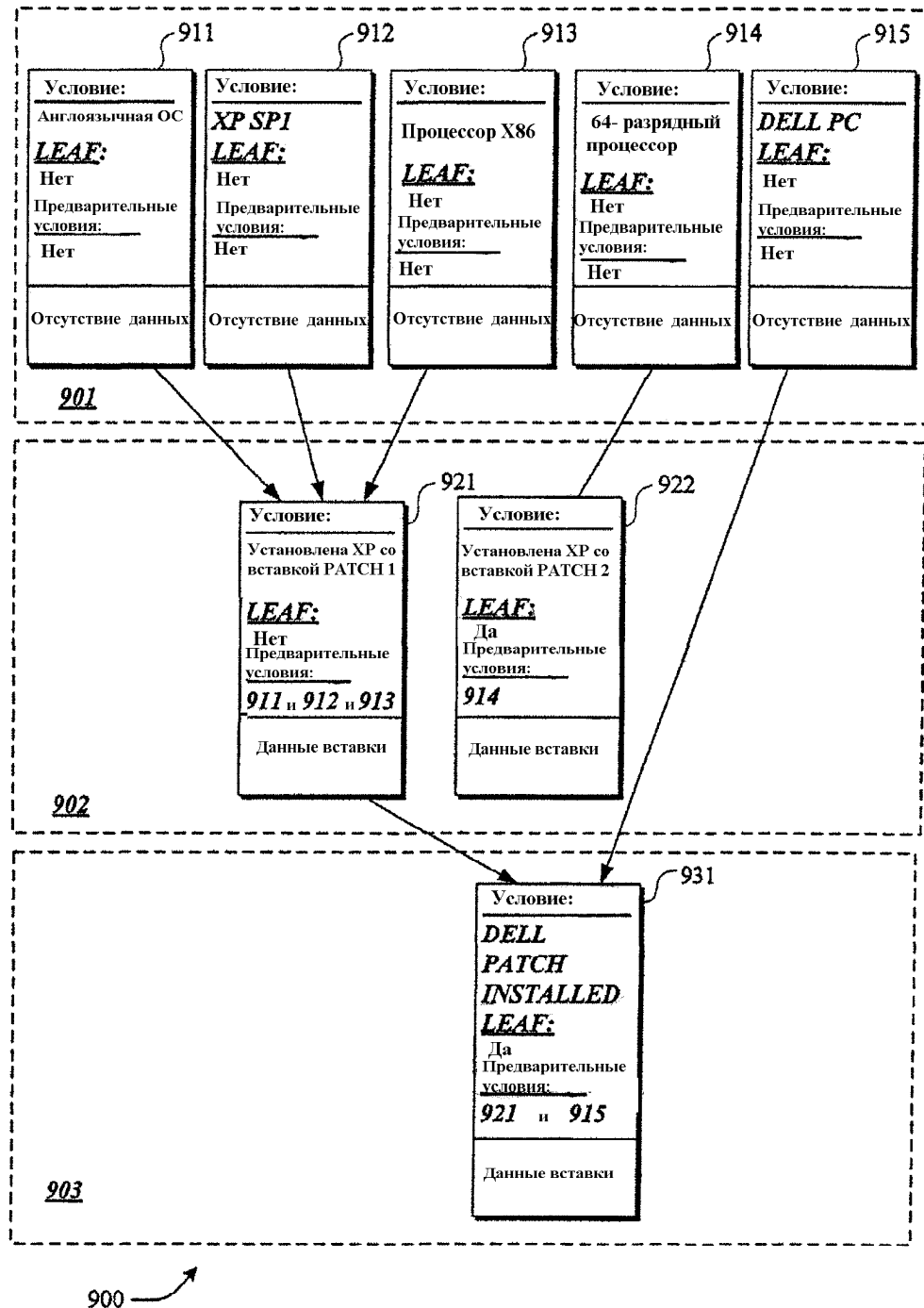
Фиг. 6



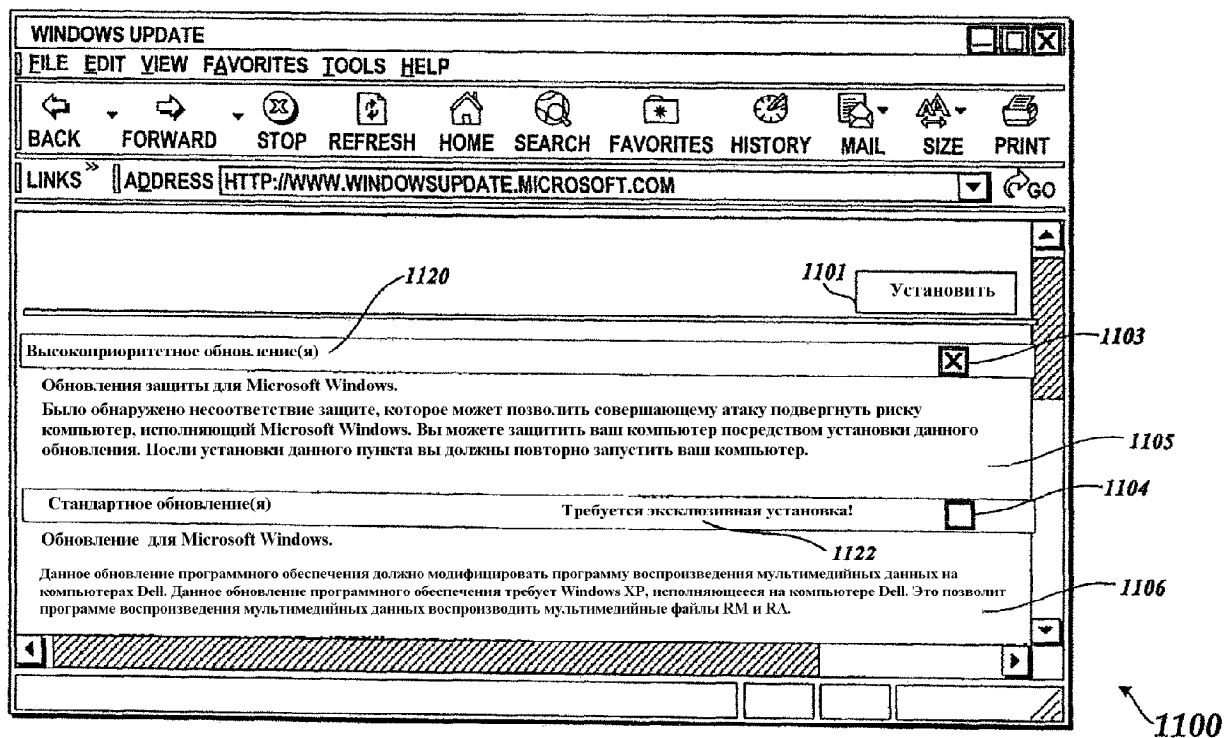
Фиг. 7



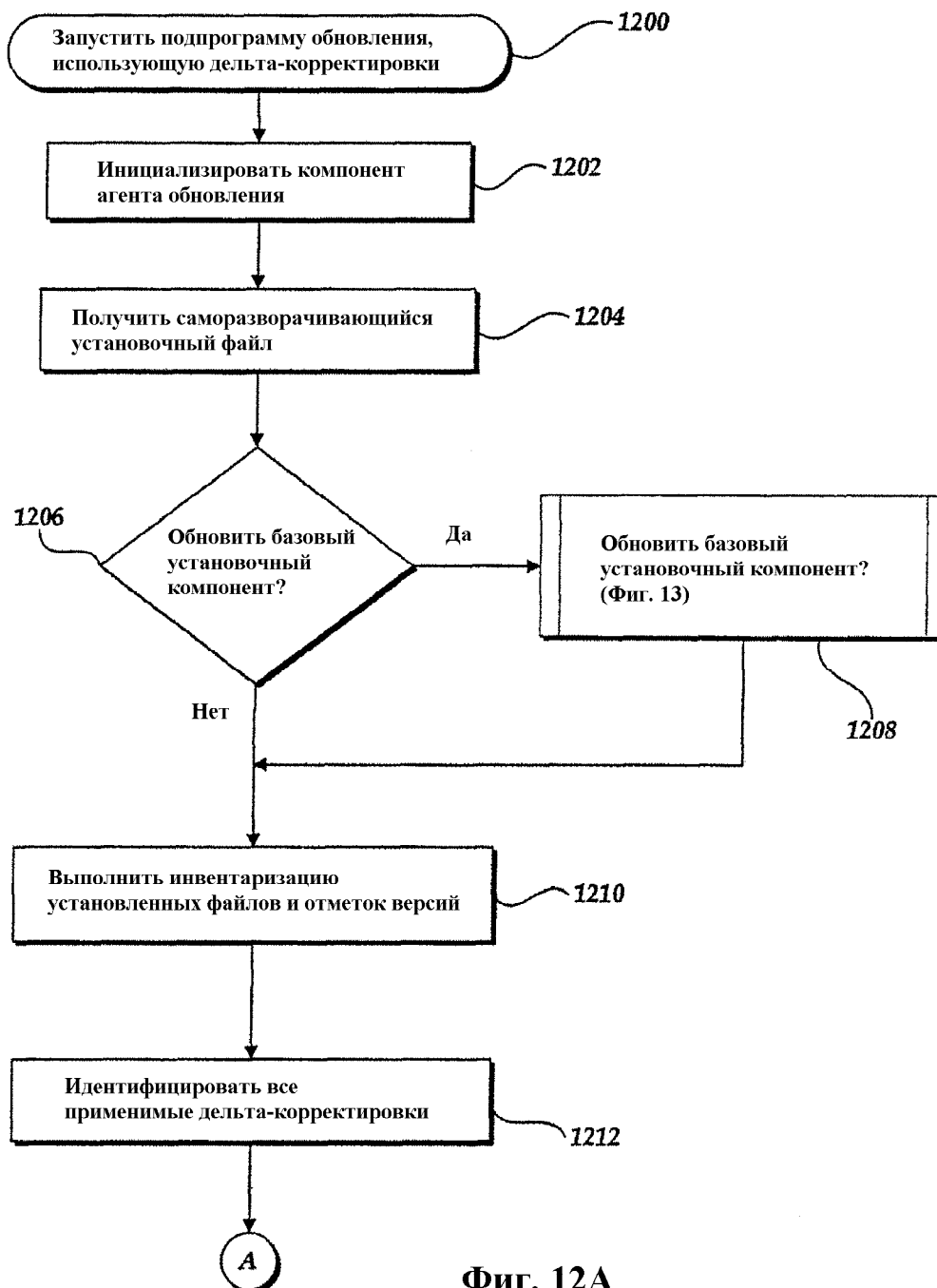
Фиг. 8



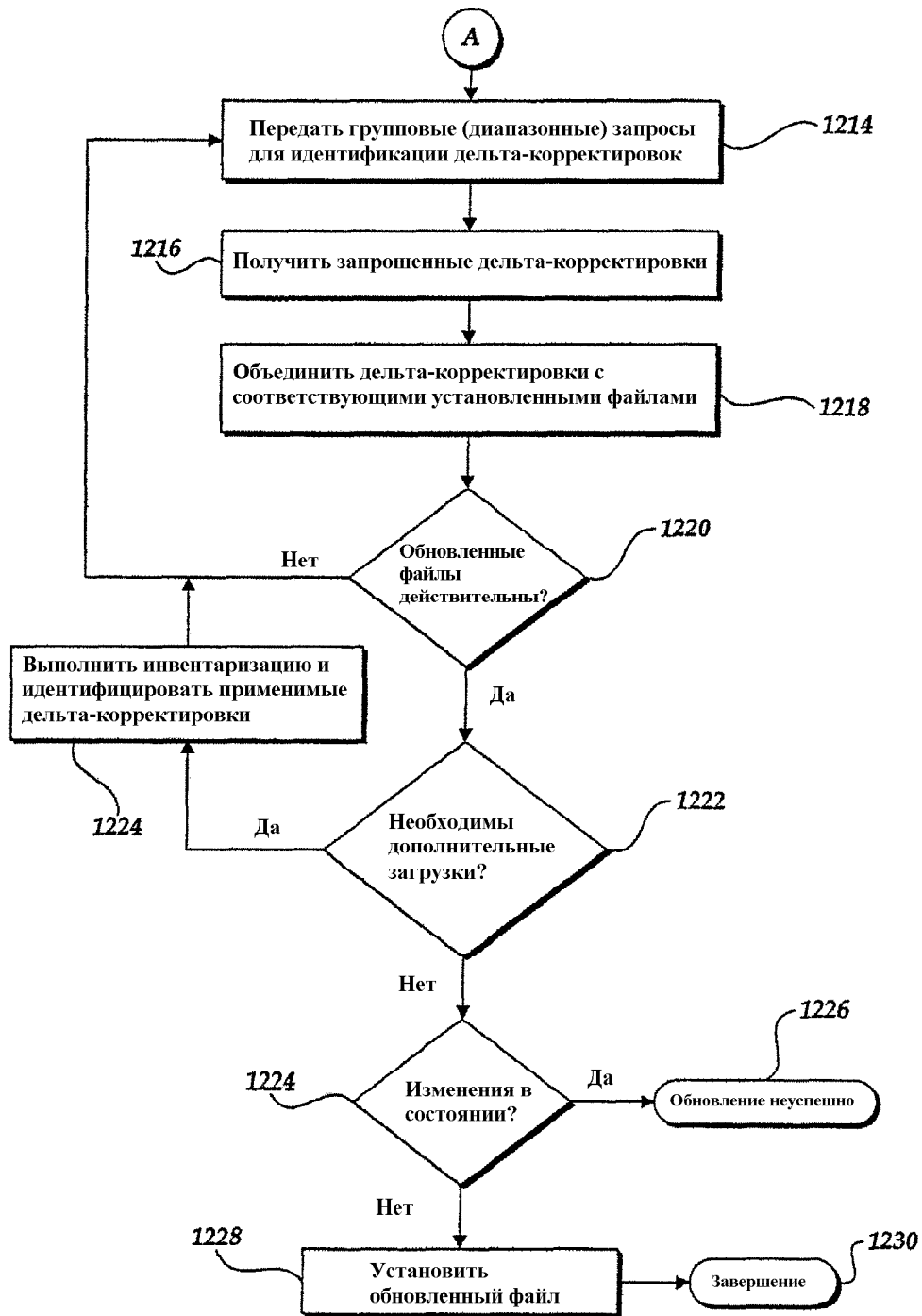
Фиг. 9



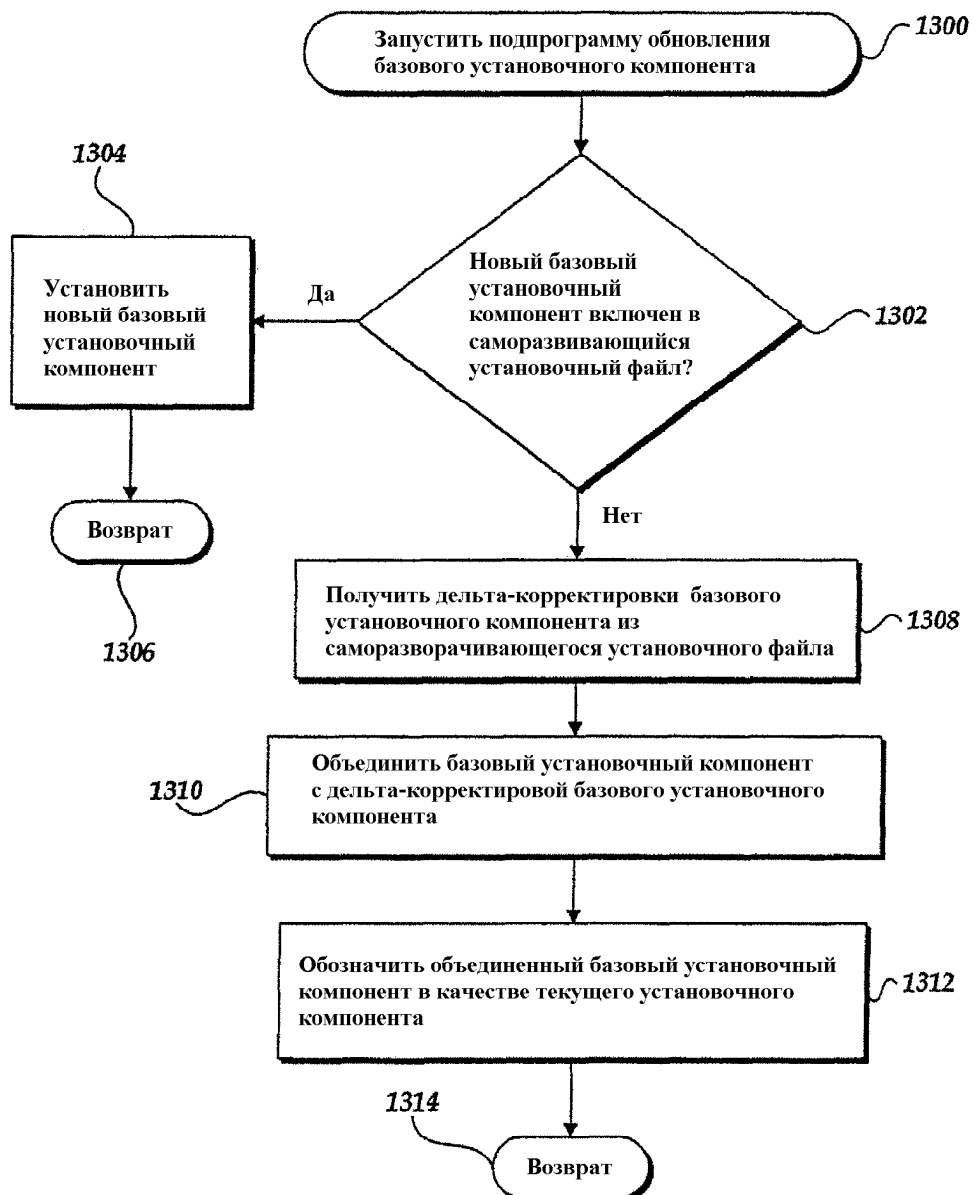
Фиг. 11



Фиг. 12А



Фиг. 12В



Фиг. 13