УДК 004.738

ИНТЕРНЕТ-СЛУЖБА ДЛЯ ПОДДЕРЖКИ МЕЖСЕРВЕРНЫХ ВЗАИМОДЕЙСТВИЙ В РАСПРЕДЕЛЕННЫХ ИНФОРМАЦИОННЫХ СИСТЕМАХ

Р. Э. Асратян

Институт проблем управления им. В. А.Трапезникова РАН, г. Москва

Описаны принципы организации межсерверных взаимодействий в Интернет-службе RFPS (службе удаленных пакетов файлов), специально предназначенной для поддержки взаимодействия компонентов распределенных управляющих систем. Отмечено, что рассмотренные механизмы взаимодействий позволяют организовать и прямой, и косвенный (через посредников) обмены пакетами между серверами с помощью одного и того же набора базисных процедур маршрутизации и одной и той же структуры данных.

ВВЕДЕНИЕ

Взаимодействия типа «сервер—сервер» приобретают все большее значение при организации распределенной обработки данных в крупномасштабных информационных системах [1]. Это связано с тем, что традиционая архитектура «клиент—сервер» часто оказывается не вполне адекватной задаче распределения «интеллекта» системы по удаленным сетевым узлам. Именно на уровне прямых контактов между серверами наиболее эффективно реализуются такие функции, как поиск удаленных компонентов распределенной системы и маршрутизация данных в сети [1, 2]. Вместе с тем специальные механизмы межсерверного взаимодействия заложены лишь в немногие из распространенных Интернет-служб, да и те ориентированы скорее на удовлетворение собственных потребностей этих служб, чем на потребности распределенных вычислений [2, 3].

В работе [4] рассматриваются принципы организации новой Интернет-службы RFPS (Remote File Packets Service — служба удаленных пакетов файлов) и, соответственно, сетевого протокола RFPP, специально ориентированных на поддержку взаимодействия компонентов в распределенных системах. Основополагающим понятием новой службы является понятие *удаленного пакета файлов,* способного перемещаться от одного узла сети к другому и аккумулировать результаты обработки на разных узлах. Служба RFPS предлагает ряд тесно связанных между собой механизмов взаимодействия, включающих в себя создание и уничтожение пакетов файлов на обслуживающих серверах, передачу данных в пакеты, извлечение данных из них и удаленную обработку загруженных в пакет данных. В данной работе рассматриваются механизмы межсерверного взаимодействия в RFPS, позволяющие организовать обмен данными между RFPS-серверами.

С точки зрения управления пакетами файлов RFPS-сервер имеет дело с объектами пяти типов:

- *клиенты* — источники запросов на обслуживание (в том числе — на открытие и закрытие пакетов);
- *обработчики* — программы (исполняемые модули), запускаемые сервером для обработки наборов данных в пакетах;
- *агенты* — постоянно активные (или периодически запускаемые) программы, объединяющие в себе свойства клиентов и обработчиков; в отличие от обработчиков, агенты могут выполняться на отдельных машинах (т. е. не на тех, на которых работает сервер и (или) клиенты); пакеты передаются на обработку агентам через связанные с агентами очереди пакетов;
- *процедуры* — подпрограммы из динамических библиотек, подключаемых к серверу для расширения его функций;
- *серверы* — другие RFPS-серверы, доступные для взаимодействия.

Объекты всех пяти типов должны быть предварительно зарегистрированы на сервере RFPS. Другими словами, сервер отвергает запросы, исходящие от незарегистрированных клиентов или агентов, или же запросы на запуск незарегистрированных обработчиков. При регистрации объекту присваивается уникальное имя; кроме того с ним связывается определенная управляющая информация (пароль, права доступа, спецификация исполняемого модуля обработчика и т. п.).

К основным формам обработки пакетов файлов относятся:

- создание и уничтожение пакета;
- заполнение пакета данными от клиента;

- применение к пакету одной или нескольких обрабатывающих программ (обработчиков), зарегистрированных на обрабатывающем сервере;
- постановка пакета в очередь на обработку к постоянно активному или периодически запускаемому агенту;
- перемещение пакета с одного сервера на другой;
- выборка клиентом результатов обработки.

Все перечисленные формы обработки можно комбинировать в разных сочетаниях. Создав удаленный пакет на каком-либо сервере RFPS, клиент может последовательно или параллельно применить к нему несколько обработчиков и (или) последовательно поставить его в очередь на обработку к нескольким агентам, и (или) переместить его на другой RFPS-сервер для продолжения обработки.

Основная цель создания RFPS — предоставить разработчику распределенной системы готовые средства сетевого взаимодействия, дав ему возможность целиком сосредоточиться на программировании обработчиков и агентов, реализующих прикладную логику.

1. ОБРАЩЕНИЕ К УДАЛЕННОМУ СЕРВЕРУ

Если для обработки пакета нужны обработчики сразу с нескольких RFPS-серверов, то возникает потребность в межсерверных взаимодействиях. Эта же потребность возникает, если необходим обмен данными между клиентами или агентами, зарегистрированными на разных RFPS-серверах. Другими словами, для этого два или более RFPS-серверов должны уметь взаимодействовать не только с собственными клиентами и агентами, но и друг с другом. Межсерверное взаимодействие основано на передаче пакетов от сервера к серверу (с «накоплением» в них результатов обработки). При передачах пакета в его управляющей информации формируется «трасса», позволяющая отследить, на каких именно серверах он «побывал» и какие именно обработчики, клиенты и агенты принимали участие в его обработке.

Все доступные для взаимодействия удаленные серверы должны быть зарегистрированы в обслуживающем сервере. Каждая регистрационная запись содержит ряд характеристик, основными из которых являются:

- регистрационное имя удаленного сервера;
- адресная информация: Интернет-имя или IP-адрес и, возможно, порт удаленного сервера;
- имя и пароль клиента, зарегистрированного на удаленном сервере, от имени которого наш сервер будет требовать обслуживания.

Как и Интернет-имена, регистрационные имена RFPS-серверов могут иметь доменную структуру, т. е. представлять собой последовательность простых имен (компонентов), разделенных точками. Если регистрационное имя в записи начинается с точки, то будут обрабатываться не один, а множество удаленных серверов (такие записи будем называть групповыми). Например, если имя имеет вид «.aux», то данная регистрационная запись будет обеспечивать доступ ко всем серверам, имена которых оканчиваются на «.aux».

С точки зрения клиента обращение к зарегистрированному удаленному серверу является частью обычной обработки, инициированной методом Process (синхронная обработка) или же парой методов Start и Wait (асинхронная обработка) [4]. Это обращение выполняется в том случае, когда в командной строке (списке имен вызываемых обработчиков, агентов и клиентов) обнаруживается специальная конструкция: «вызов удаленного сервера», имеющая следующий формат:

имя_сервера$(командная_строка)(тайм-аут),

где

- «имя_сервера» — регистрационное имя удаленного сервера;
- «командная_строка» — список имен обработчиков, агентов и клиентов на удаленном сервере (кроме имен, этот список может, в свою очередь, содержать «вложенные» вызовы удаленных серверов в качестве отдельных элементов);
- «тайм-аут» — число, отражающее максимальное время ожидания завершения обработки на удаленном сервере (в секундах).

Если список объектов содержит только один элемент, то охватывающие его скобки можно отбросить (например, alpha$zip или alpha$beta$zip). Рассмотрим пример обращения к удаленному серверу со стороны клиента:

```
RFPPClient Rfppc;
char PackId[9];
Rfppc.Config("192.168.1.77", 8126,"alibaba","sezam");
Rfppc.Connect();
Rfppc.Open(PackId);
Rfpp.SendFile(PackId,"c:\\out","file1.zip");
Rfpp.Start(PackId,"unzip;beta$trans(60);zip");
Rfpp.Wait(100);
while(Rfpp.GetFile(PackId, c:\\in", "*.*")==0);
Rfppc.Close(PackId);
Rfppc.Disconnect();
```

Первые две строки определяют две переменные: строку символов PackId и объект класса RFPPClient. Этот класс содержит в себе все методы, необходимые клиенту для работы с RFPS. Третья строка настраивает объект на работу с определенным RFPS-сервером. В качестве параметров объекту передаются IP-адрес (или имя) сервера, номер порта, имя и пароль пользователя (RFPS-сервер обслуживает только зарегистрированных пользователей).

Четвертая строка устанавливает TCP-соединение с сервером (на сервере немедленно создается отдельный процесс для обслуживания этого соединения), а пятая — открывает на сервере новый пакет: в результате выполнения метода Open строка PackId окажется заполненной уникальным (для конкретного сервера) восьмисимвольным идентификатором открытого пакета. Все последующие методы будут использовать этот идентификатор.

Шестая строка обеспечивает передачу на сервер файла из каталога «c:\out». Можно представлять себе, что файл помещается в открытый пакет. Седьмая строка начинает обработку путем вызова метода Start. В качестве аргумента передается командная строка, обеспечивающая последовательный вызов трех обработчиков. Причем первый и последний (зарегистрированные под именами «unzip» и «zip») выполняются непосредственно на обслуживающем сервере, а второй («trans») — на сервере с именем beta. Для ожидания конца обработки на удаленном сервере задается тайм-аут в 60 с.

Восьмая строка обеспечивает ожидание результатов всей обработки в течение 100 с. Девятая содержит цикл выборки всех файлов из пакета в локальный каталог клиента. Десятая и одиннадцатая обеспечивают закрытие пакета и прекращение соединения с сервером, соответственно.

Разумеется, из данного фрагмента кода намеренно удалены операторы проверки успеха выполнения методов и обработки исключений.

2. ОРГАНИЗАЦИЯ МЕЖСЕРВЕРНОГО ВЗАИМОДЕЙСТВИЯ

Обработка конструкции «вызов удаленного сервера» организуется по следующим правилам.

- Проверяется, совпадает ли «имя_сервера» с собственным именем RFPS-сервера, заданным в его конфигурации. Если совпадает, «командная_строка» обрабатывается как обычный список имен локальных объектов.
- В противном случае осуществляется поиск регистрационной записи удаленного сервера по его имени. Если запись, соответствующая имени сервера, отсутствует, то обработка пакета немедленно прекращается (с формированием соответствующего диагностического сообщения).
- Обрабатывающий пакет процесс создает динамический объект класса RFPPClient для связи с удаленным сервером. После этого он немедленно инициирует сетевое соединение, используя найденное Интернет-имя (или IP-адрес), порт, имя клиента и пароль клиента. После успешного соединения с помощью метода Open создается «пакет-клон» на удаленном сервере. В этот пакет переносятся все наборы данных и управляющая информация из обрабатываемого пакета.
- Обрабатывающий пакет процесс вызывает метод Start, инициирующий процесс обработки пакета-клона на удаленном сервере. В качестве параметра методу передается вся конструкция «вызов удаленного сервера».
- Если максимальное время ожидания обработки не задано или равно нулю, то соединение с удаленным сервером прекращается, а обработка пакета продолжается обычным образом (т. е., дальнейшая «судьба» пакета на удаленном сервере нас не интересует). В противном случае обрабатывающий пакет процесс выполняет метод Wait (с параметром, равным максимальному времени ожидания).
- После завершения обработки пакета-клона считывается и анализируется системное диагностическое сообщение, сформированное на удаленном сервере. Если оно говорит об ошибке, обработка пакета прекращается, а полученное диагностическое сообщение помещается в управляющую информацию пакета (клиент сможет получить его с помощью специального метода GetSysMsg).
- Переносятся все наборы данных и часть управляющей информации (в том числе прикладные диагностические сообщения и «трасса») из пакета-клона в обрабатываемый пакет. После этого пакет-клон уничтожается (с помощью метода Close), соединение с удаленным сервером прекращается, а обработка пакета на RFPS-сервере продолжается обычным образом.

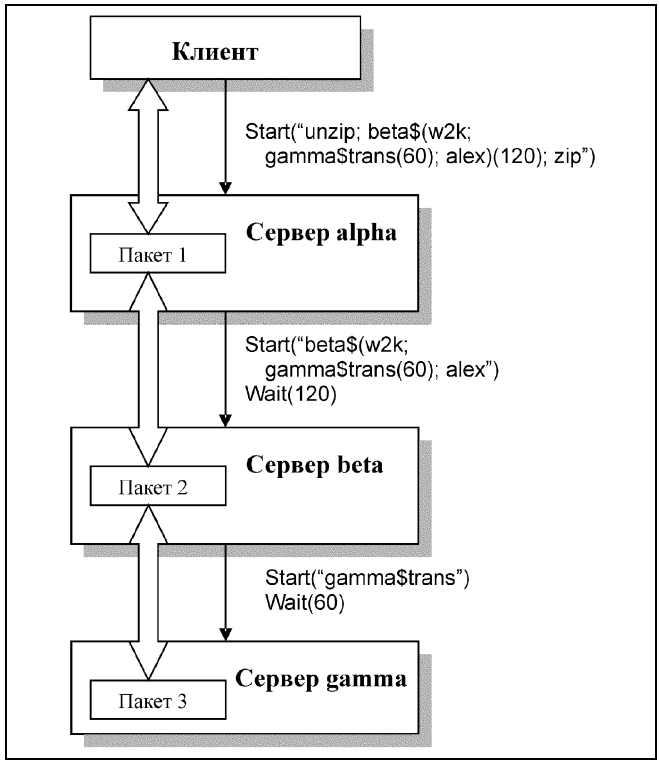

Рис. 1. Обработка вложенных вызовов удаленных серверов

Отметим, что обработка пакета-клона на удаленном сервере может, в свою очередь, вызвать обращение к другому (третьему) серверу и т. д.

Механизм обработки «вложенных» вызовов удаленных серверов проиллюстрирован на рис. 1. Предполагается, что обработчики (или агенты) unzip и zip зарегистрированы на сервере alpha, обработчик w2k и клиент alex — на сервере beta, а обработчик trans — на сервере gamma.

Как видно из рис. 1, клиент осуществляет обращение к обслуживающему его серверу alpha с помощью метода Start. В качестве параметра передается длинная командная строка, содержащая вызовы удаленных серверов (остальные параметры метода и имя объекта класса RFPPClient опущены, чтобы не загромождать рисунок). Отметим, что перед возвратом результатов на сервер alpha копия пакета будет помещена во входную очередь клиента alex на сервере beta (аналог «почтового ящика») [4].

3. МАРШРУТИЗАЦИЯ ПАКЕТОВ ФАЙЛОВ

Механизмы маршрутизации — определения направления и маршрута передачи данных — в той или иной степени используются всеми Интернет-службами. Чаще всего для решения задачи маршрутизации применяется DNS (служба имен). Однако некоторые из Интернет-служб (например, электронная почта) часто используют и собственные средства маршрутизации данных, вследствие отсутствия доступа к DNS или же недостаточности информации, предоставляемой DNS. Именно к таким службам относится и RFPS.

Даже выполнение всего одного шага из вышерассмотренного примера (например, передачи пакета от сервера alpha к серверу beta) может оказаться нетривиальной задачей, если сервер-отправитель не располагает достаточной информацией о сервере-получателе. В этом случае в полном объеме вступает в действие механизм маршрутизации пакетов.

Этот механизм позволяет избавить каждый из взаимодействующих RFPS-серверов от необходимости знать необходимые для доступа характеристики всех своих возможных «контрагентов». Сервер может воспользоваться посредником, который «знает» их вместо него и имеет соответствующие права доступа.

В связи с этим несколько уточним общую схему организации межсерверных взаимодействий, описанную в предыдущем разделе. Это уточнение касается главным образом одного пункта — поиска регистрационной записи удаленного сервера по его имени. Фактически механизм маршрутизации представляет собой правила выполнения этого поиска и интерпретации его результатов. Эти правила сводятся к следующему.

Как уже говорилось, регистрационные имена RFPS-серверов могут иметь доменную структуру, т. е. представлять собой последовательность простых имен (компонентов), разделенных точками. Как и в некоторых маршрутизаторах электронной почты, поиск регистрационной записи может осуществляться в несколько этапов. Сначала ищется запись, содержащая полное имя сервера. В случае неудачи от имени сервера «отрезается» самый левый компонент (с сохранением точки) и ищется запись, содержащая «усеченное» имя. Эта процедура повторяется до тех пор, пока не будет найдена «совпадающая» запись, или пока от имени сервера не останется ничего. В последнем случае ищется запись, содержащая одну точку в поле имени сервера. Если такая запись есть, она и считается найденной; в противном случае имя сервера признается ошибочным. Однако найденная запись вовсе не обязательно относится непосредственно к адресуемому серверу — она может относиться и к серверу-посреднику.

Интерпретация найденной записи выполняется в одном из двух режимов: «режиме запроса» (запрос адресной информации у посредников и передача пакета непосредственно адресату) или «режиме обмена» (передача через посредников). В любом случае вся обработка реализуется с помощью двух «базисных» серверных функций.

- Функция Route(name) осуществляет поиск регистрационной записи удаленного сервера по его имени (name) по вышеописанным правилам. Функция возвращает или строку, содержащую регистрационную запись со всеми атрибутами доступа (от адреса сервера до входного пароля), или сообщение об ошибке. Причем в режиме обмена функция возвращает результат и заканчивается немедленно, а в режиме запроса функция осуществляет обращение к найденному серверу, запуск в нем процедуры Route, прием строки результата по сети, возврат ее и завершение работы.
- Функция Call(name, command, access) получает в качестве аргументов имя удаленного сервера (name), командную строку (command) и строку регистрационной записи (access), доставляемую функцией Route. Функция обеспечивает соединение с удаленным сервером, описываемым строкой access и передачу ему обрабатываемого пакета вместе с именем сервера и командной строкой (см. § 2). Если переданное имя совпадает с собственным именем удаленного сервера, то в нем немедленно стартует обработка полученного пакета, заданная полученной командной строкой. В противном случае на удаленном сервере инициируется выполнение функции Call для передачи пакета на обработку в следующий сервер и т. д.

Режим обработки регистрационной строки сервера определяется меткой режима, которая может содержаться в этой строке. При отсутствии метки все групповые записи обрабатываются в режиме запроса, а негрупповые — в режиме обмена. На рис. 2. проиллюстрированы оба режима обработки на примере выполнения вызова beta$zip. Как видно из рис. 2, режим обмена основан на «рекурсивном» выполнении функции Call в цепочке серверов, а режим запроса — на «рекурсивном» выполнении процедуры Route. Разумеется, в обоих случаях имеются средства ограничения глубины рекурсии с целью предотвращения «вечного» зацикливания процесса маршрутизации.

Другими словами, в режиме запроса действия исходного сервера подчиняются логике «дайте мне информацию об удаленном сервере, чтобы я мог продолжить обработку», а в режиме обмена — «возьмите мой пакет и сделайте все сами».

На рис. 3 проиллюстрирована маршрутизация пакетов файлов на примере трех серверов (разумеется, их могло бы быть и больше), два из которых (alpha и beta) ничего друг о друге не «знают», а один (gamma) «знает» все про всех. Внутри каждого сервера в прямоугольной рамке изображен его список зарегистрированных серверов (который в данном случае фактически играет ту же роль, что и таблица маршрутизации электронной почты или таблица DNS). Каждая строка списка отображает одну регистрационную запись. В ней последовательно размещены имя сервера, его IP-адрес, а также имя и пароль для обращения к серверу. Маршрут пакета фай-

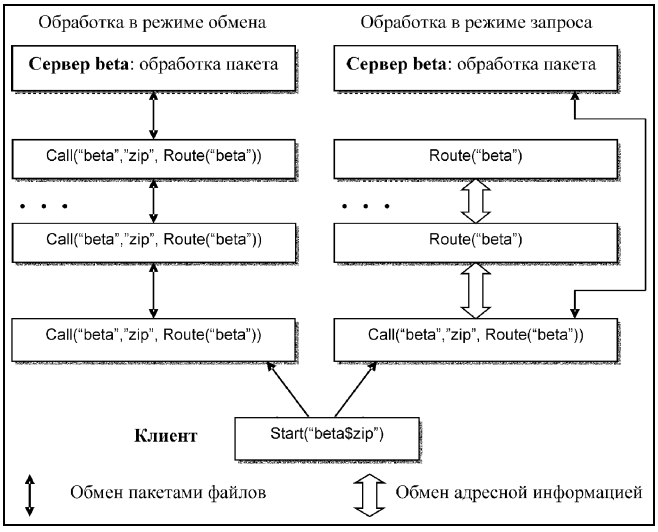

Рис. 2. Режимы маршрутизации пакетов файлов

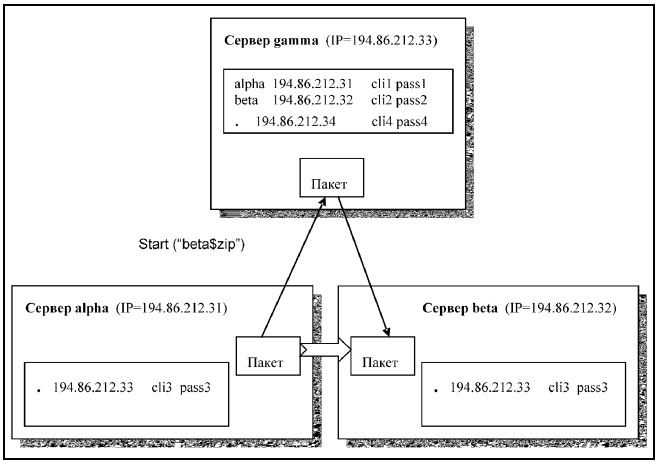

Рис. 3. Маршрутизация пакетов

лов в режиме обмена показан простыми стрелками. Как видно из рис. 3, при вызове удаленного обработчика beta$zip с сервера alpha обрабатываемый пакет будет сначала передан на сервер gamma (так как на него указывает единственная групповая запись в списке серверов). Поскольку имя адресованного сервера (beta) не совпадает с его собственным именем (gamma), он, в свою очередь, обратится к своему списку серверов для выполнения маршрутизации. В результате пакет будет передан с сервера gamma на сервер beta.

В режиме запроса маршрутизация будет проведена иначе. Сервер alpha не будет передавать пакет серверу gamma, а запросит у него данные о сервере beta. Получив эти данные (Интернет-имя или IP-адрес, порт, имя и пароль клиента для обращения к серверу), сервер alpha передаст пакет сразу на сервер beta (показано фигурной стрелкой). Режим запроса более экономичен в смысле времени и сетевого трафика, но его применение может быть затруднено «неоднородностью» сети, не допускающей прямое TCP-соединение между любыми двумя серверами.

Разумеется, в обоих режимах маршрутизация могла бы выполняться «рекурсивно». Например, если бы сервер gamma тоже испытывал недостаток маршрутной информации, он мог бы с помощью точно такого же механизма привлечь для обработки сервер delta и т. д.

В режиме запроса маршрутная информация, полученная от других серверов, может помещаться в специальное оперативное хранилище (кэш) и считаться актуальной в течение заданного администратором интервала времени (тайм-аут кэша). Если маршрутная информация об адресуемом сервере находится в кэше и еще не потеряла актуальности, то обращение к серверу организуется без каких-либо дополнительных запросов.

5. ЗАКЛЮЧЕНИЕ

Сами по себе технологии прямого и косвенного (через посредников) обмена данными между сетевыми узлами так или иначе применяются в семействе Интернет-служб (вряд ли можно придумать что-то третье). В данной работе автор попытался показать, что:

- межсерверные взаимодействия органично «вписываются» в концепцию RFPS, основанную на обработке удаленных пакетов файлов;
- режимы маршрутизации с прямой (режим запроса) и косвенной (режим обмена) передачей пакетов между серверами могут быть реализованы в рамках одной службы, с помощью одного и того же набора базисных функций и одной и той же структуры данных;
- RFPS позволяет организовать широкий спектр информационного взаимодействия: от простого обмена пакетами файлов между удаленными клиентами (аналог электронной почты) до произвольного сочетания функций передачи и обработки данных с привлечением обработчиков, агентов и клиентов, зарегистрированных на разных серверах.

Ограниченность объема статьи не позволила автору рассмотреть ряд важных механизмов RFPS, имеющих непосредственное отношение к обеспечению надежности, открытости и масштабируемости распределенных систем. К ним в первую очередь относятся:

- «альтернативные маршруты», позволяющие осуществлять обмен данными «в обход» временно неработоспособных серверов;
- «процедурная маршрутизация», позволяющая получать маршрутную информацию как результат работы зарегистрированных подключаемых процедур;
- «глобальные имена объектов», позволяющие пользователю обращаться к объектам (обработчикам, клиентам и агентам), не зная, на каких именно серверах они зарегистрированы;
- разграничение прав доступа к объектам;
- мобильные (перемещаемые в составе пакетов файлов) Java-обработчики и т. п.

В настоящее время RFPS-сервер реализован в двух вариантах: в форме сервиса для Windows2000 и в форме «демона» для Unix FreeBSD [5]. Клиентское программное обеспечение реализовано в форме библиотек функций (статической и динамической) для Win32. Первые опыты применения RFPS показали, что он легко изучаем и удобен для разработчиков и действительно способен служить «фундаментом» для межузловых информационных связей в распределенных системах.

ЛИТЕРАТУРА

1. *Асратян Р. Э., Волков А. Ф., Гаджиев А. М.* Автоматизированная система «Гибридная почта» // Труды РАН. — 1999. — Т. IV. — С. 5—16.
2. *Паркер Т.* TCP/IP. Освой самостоятельно. — М.: Бином, 1997. — 448 с.
3. *Фролов А. В., Фролов Г. В.* Глобальные сети компьютеров. — М.: Диалог-МИФИ, 1996. — 256 с.
4. *Асратян Р. Э.* Интернет-служба для поддержки распределенных информационно-управляющих систем // Проблемы управления. — 2005. — № 6. — С. 73—77.
5. *Келли-Бутл С.* Введение в Unix. — М.: ЛОРИ, 1995. — 596 с.

☎ *(495) 334-88-61*

e-mail: rea@L9.ipu.rssi.ru

❑