УДК 004.383.2

МОДЕЛЬ КЛИЕНТ-СЕРВЕРНОГО ВЗАИМОДЕЙСТВИЯ С НЕРЕЗИДЕНТНЫМИ СЕРВЕРАМИ

Д.А. КОВАЛЬ, Л.В. СЕРЕБРЯНАЯ

Белорусский государственный университет информатики и радиоэлектроники, Республика Беларусь

Поступила в редакцию 15 марта 2018

Аннотация. Разработана модель клиент-серверного взаимодействия с локальными и удаленными нерезидентными серверами, позволяющая привязать время жизни сервера к времени жизни клиента. Предложенная модель применена к расширяемому программированию приложений Microsoft Office, что позволило решить такие проблемы, как организация межмодульного взаимодействия, устранение дублирования кода и данных в каждом модуле и экспортирование общей для всех модулей функциональности.

Ключевые слова: нерезидентный сервер, модуль расширения, модуль-сервис.

Abstract. The model of client-server interaction with local and remote non-resident servers is developed, which allows to bind the server lifetime on the client lifetime. The proposed model is applied to the extensible programming of Microsoft Office applications, which allows to solve such issues like organization of cross-module interaction, elimination of code and data duplication in each module and export of common functionality for all modules.

Keywords: non-resident server, extension module, module-service.

Doklady BGUIR. 2018, Vol. 116, No. 6, pp. 12-18
The model of client-server interaction with non-resident servers
D.A. Koval, L.V. Serebryanaya

Введение

В классическом клиент-серверном взаимодействии сервер является резидентным. Это означает, что он существует (загружен в память, проинициализирован и готов к обслуживанию клиентов) вне зависимости от существования его клиентов. Работу классического клиент-серверного программного обеспечения можно описать следующей последовательностью действий.

1. Загрузка сервера в память на серверной машине.
2. Загрузка клиента в память на клиентской машине.
3. Подключение клиента к серверу.
4. Обмен информацией между клиентом и сервером.
5. Отключение клиента от сервера.
6. Выгрузка клиента из памяти на клиентской машине.
7. Завершение работы сервера (деинициализация и выгрузка из памяти на серверной машине).

Пункты 2–6 могут многократно повторяться. Таким образом, время жизни сервера никак не связано со временем жизни клиентов. Кроме того, пункты 3 и 5 актуальны только для клиент-серверного взаимодействия с установкой соединения.

В общем случае для глобальных и локальных вычислительных сетей такая модель клиент-серверного взаимодействия вполне оправдана и хорошо себя зарекомендовала.

Однако при решении задач расширяемого программирования приложений Microsoft Office потребовалось изменить общепринятый подход к клиент-серверному программированию. Данная работа посвящена задаче модификации классического клиент-серверного взаимодействия таким образом, чтобы время жизни сервера зависело от времени жизни клиентов. В предлагаемом решении сервер должен существовать, пока у него есть хотя бы один активный клиент.

Модель взаимодействия модулей расширения и модулей-сервисов

Одно из требований к расширению приложений Microsoft Office было связано с предоставлением программистам возможности создания особых модулей-сервисов. Отличие модуля-сервиса от обычного модуля расширения заключается в том, что последний загружается в адресное пространство приложения Microsoft Office, а модуль-сервис – в специальный контейнер серверов. В качестве контейнера может быть использовано выделенное виртуальное адресное пространство (процесс).

Таким образом, расширяемое программирование приложений Microsoft Office с модулями расширения и модулями-сервисами включает в себя клиент-серверное взаимодействие, где есть множество клиентов и множество серверов. Модуль расширения является клиентом, если взаимодействует с одним и более модулем-сервисом (сервером). Клиенты размещаются в процессы приложений Microsoft Office. Серверы размещаются в специальное виртуальное адресное пространство. Между клиентами и серверами прокладываются каналы взаимодействия согласно рис. 1. В случае если контейнером является выделенный процесс, то это могут быть каналы межпроцессного взаимодействия.

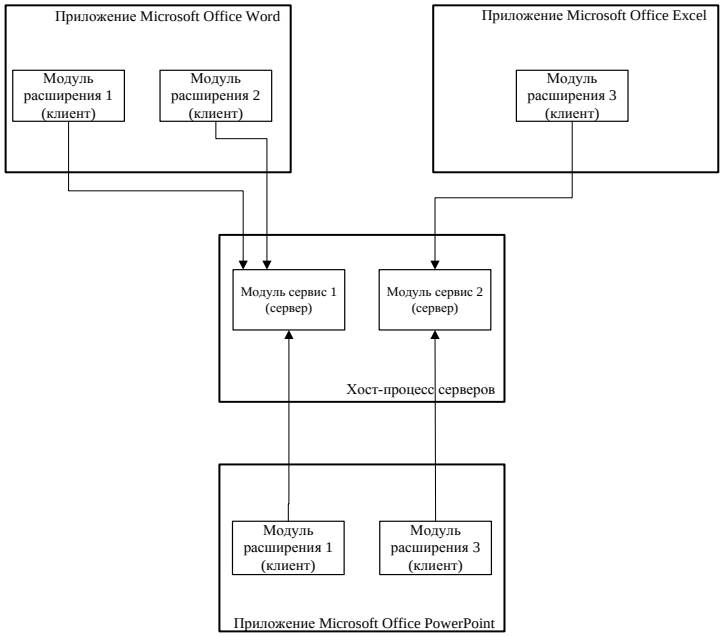

Рис. 1. Модель клиент-серверного взаимодействия модулей расширения с модулями-сервисами

Специфической особенностью такого клиент-серверного взаимодействия является то, что время жизни сервера должно зависеть от времени жизни клиентов. Если нет активных клиентов (модулей расширения в приложениях Microsoft Office), то и серверы (модули-сервисы) должны завершить свою работу (вместе с контейнером в виде процесса).

В клиент-серверных взаимодействиях можно выделить два принципиально разных типа взаимодействия: с установкой соединения и без установки соединения. К первому типу относятся взаимодействия по таким сетевым протоколам, как TCP, HTTP [1]. Ко второму типу относятся взаимодействия по сетевому протоколу UDP [2]. Кроме того, клиент-серверное взаимодействие может быть организовано с использованием таких технологий и инструментов, как RPC, COM, именованные каналы, неименованные каналы и разделяемая память (только с локальным сервером). Отслеживание активных клиентов в клиент-серверном взаимодействии

с установкой соединения сводится к задаче определения количества активных соединений и является тривиальной. В случае взаимодействия без установки соединения ситуация гораздо сложнее и готового решения пока не существует.

Предоставляя интерфейсы прикладного программирования для разработки модулей-сервисов, нельзя ограничиваться возможностью создания только серверов, работа с которыми требует установки соединения. Более того, специфика предметной области способствует организации клиент-серверного взаимодействия, оптимизированного под взаимодействие с локальными серверами. Это позволяет увеличить пропускную способность и уменьшить накладные расходы при передаче сообщений между клиентом и сервером. Клиент-серверное взаимодействие с локальными серверами может исключить из стека взаимодействия такой модуль операционной системы, как сеть (драйверы стека TCP/IP, HTTP и так далее) [3].

Для решения ранее поставленной задачи будем рассматривать множество клиентов, взаимодействующих с множеством серверов. В качестве клиентов в работе приняты модули расширения, а в качестве серверов – модули-сервисы. Модель взаимодействия должна позволять привязать время жизни сервера к времени жизни клиентов и учитывать тот факт, что клиенты могут взаимодействовать с серверами с помощью различных технологий – как с установкой соединения, так и без установки соединения.

Алгоритм взаимодействия клиентов с локальными и удаленными нерезидентными серверами

В основу решения задачи по организации механизма взаимодействия клиентов с нерезидентными серверами легли понятия виртуальный сервер, виртуальный клиент и виртуальное соединение. Виртуальный сервер – это специальный системный модуль-сервис, не содержащий бизнес-логики. Единственное назначение такого модуля-сервиса – управление временем жизни контейнера серверов. Виртуальный клиент – это специальный системный модуль расширения, задача которого состоит в удаленном управлении временем жизни виртуального сервера посредством виртуального соединения. Виртуальное соединение – средство управления временем жизни виртуального сервера, которое используется виртуальными клиентами. На рис. 2 представлена модель взаимодействия клиентов с нерезидентными серверами. Ее можно описать следующим алгоритмом.

1. В память загружается хост-процесс (приложение Microsoft Office) для клиентов (модулей расширения).
2. В память приложения Microsoft Office загружается виртуальный клиент.
3. Виртуальный клиент проверяет наличие виртуального сервера. В случае его отсутствия контейнер для серверов вместе со всеми серверами (модулями-сервисами), включая виртуальный, загружаются в память.
4. Виртуальный клиент устанавливает виртуальное соединение с виртуальным сервером. При этом сервер регистрирует клиента в своем реестре.
5. В память приложения Microsoft Office загружаются все модули расширения этого приложения (клиенты).
6. Виртуальный клиент периодически посылает специальное системное сообщение виртуальному серверу с установленным интервалом времени, тем самым давая понять серверу, что он (клиент) все еще активен.
7. Хост-процесс клиентов (приложение Microsoft Office) завершает свою работу.
8. Если приложение Microsoft Office завершает свою работу в нормальном режиме, то виртуальный клиент отключается от виртуального сервера, при этом виртуальный сервер удаляет его из своего реестра.
9. Если приложение Microsoft Office завершает свою работу в аварийном режиме, виртуальный клиент может не успеть закрыть виртуальное соединение. Если через установленный интервал времени сервер не получает сообщение от клиента, то виртуальный сервер удаляет виртуального клиента из своего реестра автоматически (разрыв виртуального соединения).
10. Если реестр виртуального сервера не содержит ни одной записи, виртуальный сервер завершает свою работу, завершая работу контейнера серверов.

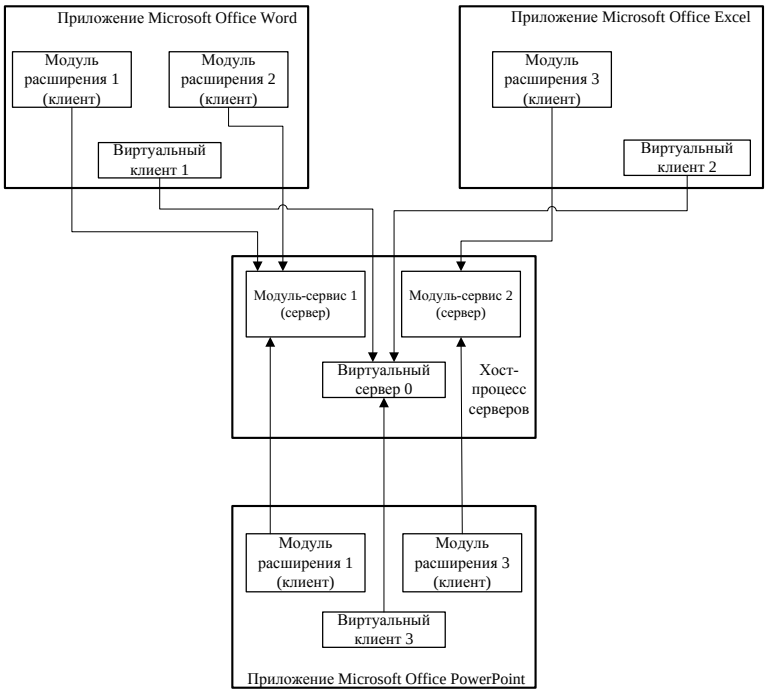

Рис. 2. Модель клиент-серверного взаимодействия модулей расширения с нерезидентными модулями-сервисами

Предложенный алгоритм позволяет абстрагироваться от конкретной реализации клиент-серверного взаимодействия. Это, в свою очередь, дает возможность программировать клиент-серверное взаимодействие, используя как технологии с установкой соединения, так и технологии без установки соединения. Данная модель позволяет организовать клиент-серверное взаимодействие как с локальными, так и с удаленными нерезидентными серверами. Очевидно, что платой за нерезидентность сервера является увеличение трафика между клиентами и сервером на величину количества системных сообщений, посылаемых от клиентов к серверу с целью уведомить сервер о том, что клиент все еще активен.

Обобщенность модели на локальные и глобальные вычислительные сети, а также на локальные и удаленные сервера учитывает тот факт, что инициатором взаимодействия может быть только клиент. Это обусловлено следующим.

1. Архитектурой глобальных вычислительных сетей, которые построены по принципу агрегации локальных вычислительных сетей с использованием технологии NAT [4]. Суть ее заключается в том, что глобальный IP-адрес имеет только один узел (шлюз) из локальной вычислительной сети, а все остальные узлы сети имеют только локальные IP-адреса. Технология NAT основана на разделении портов шлюза на два диапазона – младший и старший. Младший диапазон портов используется самим шлюзом, а старший отведен для реализации технологии NAT. Сетевой пакет, предназначенный для вычислительного узла, находящегося за пределами локальной сети отправителя, поступает на шлюз. Шлюз, получив такой сетевой пакет, заменяет в нем IP-адрес и порт отправителя на свой глобальный IP-адрес и порт из старшего диапазона портов соответственно, делая соответствующую запись в своем реестре NAT. После этого шлюз пересылает модифицированный сетевой пакет в глобальную вычислительную сеть. При приеме сетевых пакетов из глобальной вычислительной сети шлюз определяет порт получателя. Если порт относится к старшему диапазону, то шлюз использует этот порт как ссылку на пару IP-адрес и порт в своем реестре NAT. Найдя IP-адрес и порт получателя в реестре NAT, шлюз заменяет в сетевом пакете IP-адрес и порт получателя на найденные в реестре. После этого шлюз отправляет сетевой пакет в соответствующую локальную вычислительную сеть. Таким образом, сервер не может быть инициатором взаимодействия по причине сокрытия реального адреса клиента.

2. Принципами безопасности (сетевые экраны, брандмауэры и прочее).

Алгоритм клиент-серверного взаимодействия с локальными нерезидентными серверами

Учитывая специфику задачи расширяемого программирования приложений Microsoft Office, а именно то, что серверы являются локальными, приведенную выше модель можно оптимизировать, как показано на рис. 3. Ниже приведен соответствующий модели алгоритм.

1. В память загружается хост-процесс (приложение Microsoft Office) для клиентов (модулей расширения).
2. В память приложения Microsoft Office последовательно загружаются виртуальный сервер и виртуальный клиент.
3. Виртуальный клиент приложения Microsoft Office проверяет наличие виртуального сервера хост-процесса серверов. В случае его отсутствия контейнер для серверов вместе со всеми серверами (модулями-сервисами), включая виртуальный сервер и виртуального клиента, загружаются в память.
4. Виртуальный клиент приложения Microsoft Office устанавливает виртуальное соединение с виртуальным сервером хост-процесса серверов. При этом виртуальный сервер регистрирует виртуального клиента в своем реестре.
5. Для каждой регистрации виртуального клиента в хост-процессе серверов виртуальный клиент этого процесса устанавливает виртуальное соединение с виртуальным сервером приложения Microsoft Office. При этом виртуальный клиент регистрирует виртуальный сервер в своем реестре.
6. В память приложения Microsoft Office загружаются все модули расширения этого приложения (клиенты).
7. Виртуальный клиент хост-процесса серверов выбирает любой виртуальный сервер приложений Microsoft Office (из своего реестра) и проверяет, существует ли он. Алгоритм выбора может быть любым – например, серверы могут выбираться по очереди из списка или случайным образом.
8. Если виртуальный сервер существует, то предыдущий шаг повторяется через заданный интервал времени.
9. Если виртуальный сервер не существует, то он удаляется из реестра серверов, и осуществляется переход на 7-й шаг.
10. Хост-процесс клиентов (приложение Microsoft Office) завершает свою работу.
11. Если приложение Microsoft Office завершает свою работу в нормальном режиме, то виртуальный клиент в приложении Microsoft Office отключается от виртуального сервера хост-процесса серверов, при этом виртуальный сервер хост-процесса серверов удаляет его из своего реестра.
12. Если приложение Microsoft Office завершает свою работу в аварийном режиме, виртуальный клиент приложения Microsoft Office может не успеть закрыть виртуальное соединение. Если через установленный интервал времени виртуальный сервер хост-процесса серверов не получает сообщение от виртуального клиента приложения Microsoft Office, то виртуальный сервер удаляет виртуального клиента из своего реестра автоматически (разрыв виртуального соединения).
13. Если реестр клиентов или реестр серверов в хост-процессе серверов не содержит ни одной записи, то виртуальный сервер или виртуальный клиент соответственно завершает свою работу, завершая работу контейнера серверов.

Предложенные модель и алгоритм позволяют значительно уменьшить трафик между клиентами и локальными серверами. Вместо того чтобы каждый виртуальный клиент приложений Microsoft Office отправлял системное сообщение виртуальному серверу хост-процесса серверов, в хост-процесс серверов внедряется виртуальный клиент для опроса статуса приложений Microsoft Office, в которые внедрены виртуальные сервера. Причем для продолжения работы контейнера серверов достаточно знать о том, что хоть один виртуальный сервер приложений Microsoft Office существует. Реестр виртуальных серверов в хост-процессе серверов может какое-то время содержать мусор в виде записей о завершивших работу серверах, однако это допустимо, поскольку критерий работы – одна и более запись об активном сервере. Сборка мусора происходит время от времени при выяснении статуса активности виртуальных серверов: неактивный сервер удаляется из реестра и выбирается следующий для выяснения статуса, пока не найден первый активный сервер.

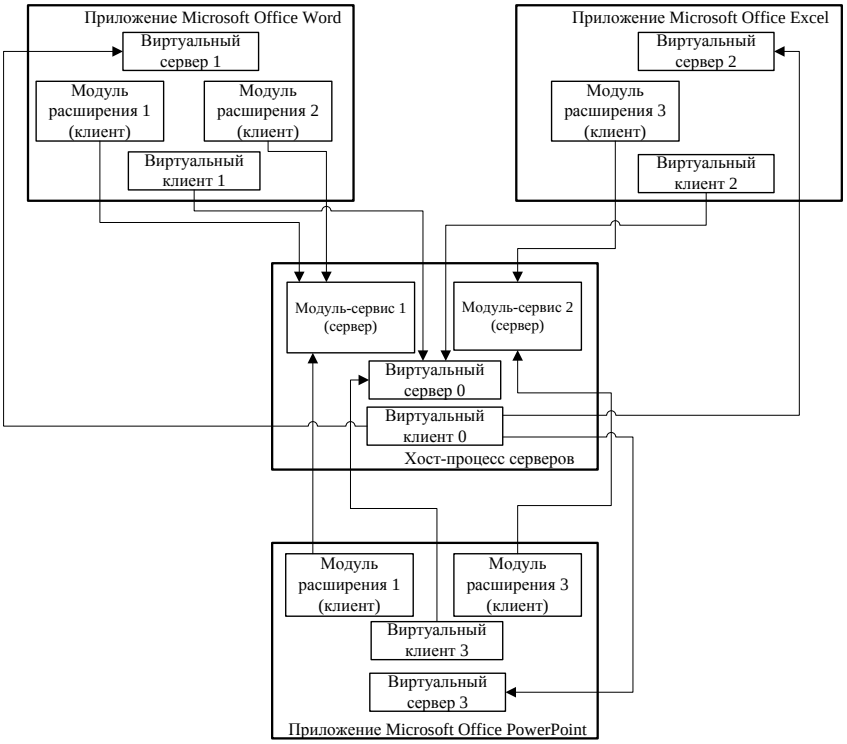

Рис. 3. Модель клиент-серверного взаимодействия модулей расширения с локальными нерезидентными модулями-сервисами

Заключение

Таким образом, разработанная модель клиент-серверного взаимодействия с нерезидентными серверами может быть успешно применена для расширяемого программирования приложений Microsoft Office, в частности, для контроля времени жизни модулей-сервисов и их контейнера.

Модули-сервисы позволяют решить важнейшие задачи, стоящие в области расширяемого программирования приложений Microsoft Office:

1. Предоставление общей для всех модулей расширения функциональности.
2. Устранение дублирования кода и данных в каждом модуле расширения.
3. Организация взаимодействия модулей расширения между различными хост-приложениями.

Примером модуля-сервиса с общей для всех модулей функциональностью может быть сервис аутентификации. Он способен инкапсулировать в себе логику аутентификации, предоставляя потребителям лишь интерфейс. Модуль-сервис по управлению пользовательскими интерфейсами, отличными от интерфейсов приложений Microsoft Office, может служить примером устранения дублирования кода и данных в каждом модуле. Межмодульное взаимодействие для модулей расширения, находящихся в различных приложениях Microsoft Office, необходимо в случае, если эти модули решают какую-то общую прикладную задачу.

Важность разработанной модели заключается в том, что она позволяет решить задачу расширяемого программирования приложений Microsoft Office модулями-сервисами, контролируя их время жизни. Существующие модели клиент-серверного взаимодействия не решают проблему контроля времени жизни сервера либо решают ее в определенных условиях (взаимодействие с установкой соединения). Применение разработанной модели к расширяемому программированию приложений Microsoft Office позволяет связать время жизни хост-процесса модулей-сервисов (серверов) со временем жизни приложений Microsoft Office (клиентов).

Список литературы

1. Fall K.R., Stevens W.N. TCP/IP Illustrated. Vol. 1: The Protocols. Boston; Addison-Wesley Professional, 2011. 1056 p.
2. Asif M. UDP/IP For Embedded System: Methods, Implementation, Benchmarks, Programming, FPGA, Embedded system. LAP LAMBERT Academic Publishing, 2010. 104 p.
3. Руссинович М., Соломон Д. Внутреннее устройство Microsoft Windows. Питер, 2013. 800 с.
4. Gibson D. Microsoft Windows Networking Essentials. San Francisco; Sybex, 2011. 372 p.

References

1. Fall K.R., Stevens W.N. TCP/IP Illustrated. Vol. 1: The Protocols. Boston; Addison-Wesley Professional, 2011. 1056 p.
2. Asif M. UDP/IP For Embedded System: Methods, Implementation, Benchmarks, Programming, FPGA, Embedded system. LAP LAMBERT Academic Publishing, 2010. 104 p.
3. Russinovich M., Solomon D. Vnutrennee ustrojstvo Microsoft Windows. Piter, 2013. 800 s. (in Russ.)
4. Gibson D. Microsoft Windows Networking Essentials. San Francisco; Sybex, 2011. 372 p.

Сведения об авторах

Коваль Д.А., аспирант кафедры программного обеспечения информационных технологий Белорусского государственного университета информатики и радиоэлектроники.

Серебряная Л.В., к.т.н., доцент кафедры программного обеспечения информационных технологий Белорусского государственного университета информатики и радиоэлектроники.

Адрес для корреспонденции

220013, Республика Беларусь
г. Минск, ул. П. Бровки, 6
Белорусский государственный университет
информатики и радиоэлектроники
тел. +375-29-119-36-44;
e-mail: dimko6669@yandex.ru
Коваль Дмитрий Александрович

Information about the authors

Koval D.A., master of technical science, PG student of information technology software department of Belarusian state university of informatics and radioelectronics.

Serebryanaya L.V., PhD, associate professor of information technology software department of Belarusian state university of informatics and radioelectronics.

Address for correspondence

220013, Republic of Belarus,
Minsk, P. Brovka st., 6,
Belarusian state university
of informatics and radioelectonics
tel. +375-29-119-36-44;
e-mail: dimko6669@yandex.ru
Koval Dzmitry Aleksandrovich