УДК 004.912: 004.822
DOI: 10.17586/0021-3454-2023-66-2-112-117

ОСОБЕННОСТИ РЕАЛИЗАЦИИ РАСПРЕДЕЛЕННОЙ ВИРТУАЛЬНОЙ МАШИНЫ ПРИ ПОСТРОЕНИИ КОММУНИКАЦИОННОЙ ИНФРАСТРУКТУРЫ ОБЩЕГО НАЗНАЧЕНИЯ

С. В. Кулешов, И. О. Шальнев*

Санкт-Петербургский федеральный исследовательский центр Российской академии наук
Санкт-Петербург, Россия
**shalnev.i@iias.spb.su*

Аннотация. Решается задача построения архитектуры распределенной виртуальной машины. Предложена техническая реализация распределенной виртуальной машины, обеспечивающей построение инфокоммуникационной инфраструктуры на новых принципах. Передача данных между узлами сети при этом происходит как программное взаимодействие между отдельными узлами виртуальной машины, что позволяет разработчику приложений в рамках распределенной системы абстрагироваться от коммуникационного уровня. Предложены минимально достаточный набор инструкций виртуальной машины для обеспечения удаленного вызова функций, а также механизм расширения набора инструкций прикладными функциями. Отличительной особенностью архитектуры является использование объектно-ориентированного подхода для построения расширяемой среды выполнения байт-кода с возможностью обращения к объектам, расположенным удаленно на узлах сети. Подход применим в ряде практических задач, в частности, для построения распределенных интерактивных приложений, информационных систем, а также для организации коммуникации между робототехническими комплексами в роевых сценариях их применения.

Ключевые слова: *виртуальная машина, транспортный пакет, инфокоммуникация, байт-код, интерпретатор*

Благодарности: работа поддержана Государственным заданием № FFZF-2022-0005.

Ссылка для цитирования: *Кулешов С. В., Шальнев И. О.* Особенности реализации распределенной виртуальной машины при построении коммуникационной инфраструктуры общего назначения // Изв. вузов. Приборостроение. 2023. Т. 66, № 2. С. 112—117. DOI: 10.17586/0021-3454-2023-66-2-112-117.

FEATURES OF THE IMPLEMENTATION OF A DISTRIBUTED VIRTUAL MACHINE WHEN BUILDING A GENERAL-PURPOSE COMMUNICATION INFRASTRUCTURE

S. V. Kuleshov, I. O. Shalnev*

St. Petersburg Federal Research Center of the RAS,
St. Petersburg, Russia
shalnev.i@iias.spb.su

Abstract. The problem of building the architecture of distributed virtual machine is being solved. A technical implementation of a distributed virtual machine is proposed, which provides the construction of an infocommunication infrastructure based on new principles. In the case under consideration, data transfer between network nodes occurs as a software interaction between individual nodes of a virtual machine, which allows the application developer within a distributed system to abstract from the communication level. A minimally sufficient set of instructions for a virtual machine is proposed to provide remote function calls, as well as a mechanism for extending the set of instructions with application functions. A distinctive feature of the architecture is the use of an object-oriented approach to build an extensible bytecode execution environment with the ability to access objects located remotely on network nodes. The approach is applicable in a number of practical tasks, in particular, for building distributed interactive applications, information systems, as well as for organizing communication between robotic systems in swarm scenarios of their use.

Keywords: virtual machine, transport packet, infocommunication, bytecode, interpreter

Acknowledgment: the work was supported by State Assignment No. FFZF-2022-0005.

For citation: Kuleshov S. V., Shalnev I. O. Features of the implementation of a distributed virtual machine when building a general-purpose communication infrastructure. *Journal of Instrument Engineering.* 2023. Vol. 66, N 2. P. 112—117 (in Russian). DOI: 10.17586/0021-3454-2023-66-2-112-117.

Одним из эффективных способов реализации коммуникационной инфраструктуры общего назначения для передачи цифровых данных в сети терминальных устройств является использование распределенной виртуальной машины (РВМ) [1]. Как было отмечено в [1], коммуникационная среда для межкомпонентного взаимодействия реализуется при этом средствами самой виртуальной машины.

Такой подход имеет следующие преимущества:

— используя средства распределенной виртуальной машины, разработчик приложений получает все преимущества коммуникационной инфраструктуры, при этом не затрачивая ресурсы на ее разработку;

— изменение функционального содержания программных компонентов узлов сети возможно без обновления их программного кода.

Традиционно технология виртуализации применялась для разделения ресурсов между задачами или пользователями, а также для реализации аппаратных ресурсов программно [2]. Виртуальные машины как основной механизм технологии виртуализации обеспечивают программную реализацию вычислительного устройства с заданными свойствами [3]. Специфика применения таких машин накладывает требования и ограничения на техническую реализацию для обеспечения потребностей инфокоммуникации. Рассмотрим эти особенности подробнее.

Распределенная виртуальная машина — программная реализация совокупности взаимодействующих языковых виртуальных машин, поддерживающих парадигму объектно-ориентированного подхода [4]. Эти машины работают в едином адресном пространстве и в общем пространстве имен объектов и функций, они обеспечивают доступ к функциям удаленных объектов. Возможная структура РВМ приведена на рис. 1.

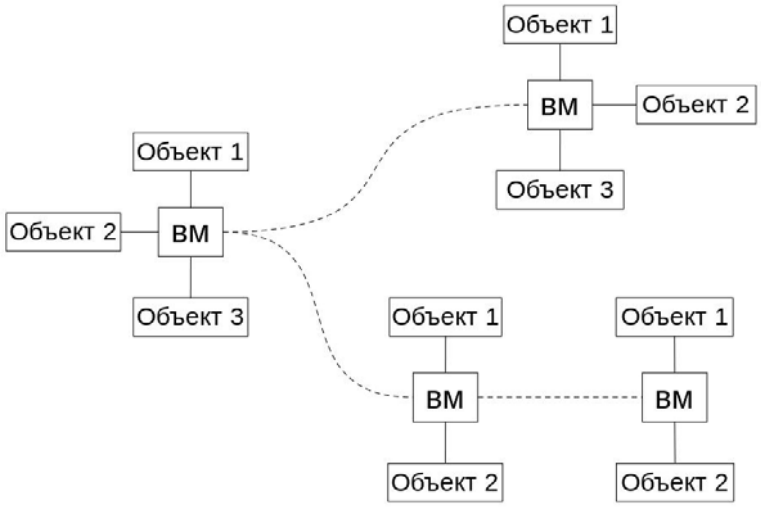

Рис. 1

Изменяя расположение объектов по узлам РВМ, можно обеспечить:

— расположение объекта (т.е. исполнимого программного кода, соответствующего данному объекту) непосредственно на узле, на котором он функционирует (в первую очередь речь идет о периферийных устройствах, содержащихся в узлах, таких как датчики и исполнительные механизмы);

— повышение эффективности работы узлов за счет балансировки объема исполняемого кода между узлами РВМ, с учетом оценки доступных вычислительных ресурсов каждого узла.

Основным недостатком при этом является некоторое возрастание нагрузки на физические каналы передачи данных вследствие увеличения объема передаваемых данных. Это происходит в связи с тем, что вместе с данными, относящимися к бизнес-логике приложения, требуется передавать фрагменты исполнимого кода, имплементирующие объекты на удаленных узлах РВМ.

Распределенное взаимодействие [5] между узлами сети обеспечивается при помощи механизма удаленного вызова методов (Remote Method Invocation, RMI) [6, 7]. Функционирование узлов РВМ предполагает выполнение программного кода вычислителями (процессорами) этих узлов — интерпретации инструкций байт-кода. Интерпретатор реализован в функции, где началу каждой инструкции сопоставляется адрес в памяти (метка), к которой можно обратиться при помощи оператора безусловного перехода (оператор goto). Внутреннее исполнение байт-кода таким интерпретатором представлено в виде последовательности меток адресов памяти на начало инструкций. Примером подобного способа представления байт-кода является Direct Threading Table (DTT), используемый в виртуальной машине JVM [8] языка программирования Java (рис. 2).

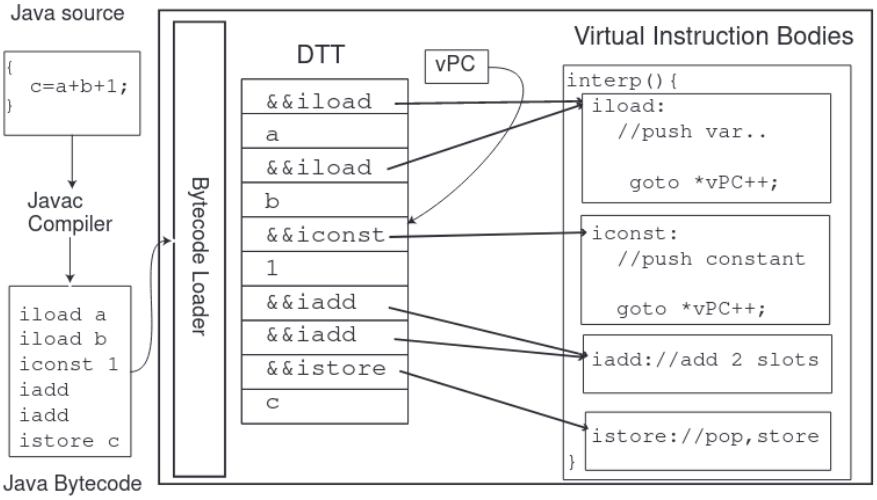

Рис. 2

Интерпретация байт-кода начинается с установки указателя vPC на адрес первой метки в DTT, после которого происходит „переход“ на адрес, хранящийся в таблице. Каждое тело инструкции заканчивается оператором goto *vPC++, который инкрементирует счетчик операций, указывающий на следующую инструкцию, находящуюся в загруженном представлении байт-кода.

Собственно коммуникационное взаимодействие внутри РВМ реализуется на уровне абстракции класса Connection, который инкапсулирует все основные функциональные блоки взаимодействия вычислительных узлов, скрывая от разработчика технические подробности сетевого взаимодействия. Класс Connection содержит функциональные блоки:

— управления транспортным слоем;
— формирования и подготовки транспортных пакетов;
— сборки пакета байт-кода (Script Assembler);
— управления коллекциями классов аксессоров;
— управления таблицами API и аксессор-объектов;
— управления языковой виртуальной машиной.

Ключевым блоком РВМ является языковая виртуальная машина, которая интерпретирует байт-код, включающий четыре основных инструкции [9]:

— PUSH — инструкция размещения значения в верхушку стека;
— CRT — инструкция создания распределенного объекта;
— DEL — инструкция удаления распределенного объекта;
— CALL — инструкция вызова метода распределенного объекта.

Каждая инструкция сопровождается аргументами, размещаемыми в стеке при помощи инструкции PUSH. Инструкция CRT использует четыре аргумента или более (рис. 3). Первые два — идентификаторы коллекции (библиотеки функций) и типа объекта, третий аргумент — непосредственно идентификатор, позволяющий однозначно определить удаленный объект в

среде выполнения РВМ. При создании объекта виртуальная машина добавляет соответствующую запись в таблицу, а по идентификатору она может обращаться к области памяти, в которой этот объект создан и локализован. Кроме того, таблица объектов является средством синхронизации вычислительных устройств на узлах распределенной системы и находится как на стороне подлинного объекта (области кода, имплементирующего объект), так и на стороне объекта-заглушки (кода, реализующего интерфейс к подлинному объекту). Именно при помощи этой структуры данных возможно манипулировать объектами в РВМ.

Четвертый аргумент инструкции CRT является идентификатором конструктора для исполняемого объекта, так как в общем случае конструкторов у объекта может быть несколько. Опциональные (не обязательные) аргументы, начиная с пятого, относятся непосредственно к методу-конструктору исполняемого объекта.

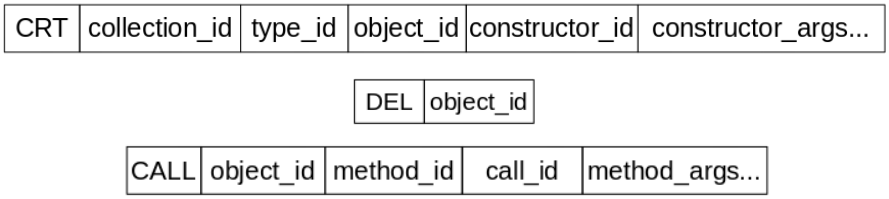

Рис. 3

Инструкция удаления объекта DEL имеет один аргумент — идентификатор удаляемого объекта. При исполнении инструкции DEL объект будет удален как из памяти, так и из таблицы объектов.

Инструкция CALL производит вызов удаленного метода. Первый аргумент инструкции — идентификатор объекта, к которому выполняется обращение, второй аргумент — идентификатор метода выбранного объекта. Третий аргумент служит для реализации механизма обратного вызова (callback function), чтобы обеспечить возможность асинхронного взаимодействия, и содержит идентификатор метода, который будет вызван при завершении функции. Опциональные аргументы, начиная с четвертого, относятся непосредственно к вызываемому методу и передаются ему в качестве аргументов.

Описанное множество инструкций необходимо и достаточно для организации распределенного взаимодействия в рамках РВМ. Идентификаторы методов являются механизмом расширения системы инструкций виртуальной машины, при этом каждая пользовательская или прикладная инструкция, вновь вводимая программистом, необходима для вызова исполнимого кода какого-либо объекта. Таким образом, добавлением нового объекта в пул объектов РВМ расширяются система инструкций данной РВМ и соответственно интерпретатор виртуальной машины.

Выводы. В статье предложен вариант технической реализации распределенной виртуальной машины, обеспечивающей построение инфокоммуникационной инфраструктуры на новых принципах. Передача данных между узлами сети при этом происходит как программное взаимодействие отдельных узлов виртуальной машины, что позволяет разработчику приложений в рамках распределенной системы абстрагироваться от коммуникационного уровня.

Раскрыты основные особенности предложенной системы инструкций, а также механизма расширения набора функций языковой виртуальной машины прикладными и пользовательскими функциями.

Использование такого подхода позволяет повысить скорость разработки распределенных программных информационных и информационно-управляющих систем и увеличить эффективность их работы за счет балансировки объема исполняемого кода между узлами распределенной виртуальной машины, основанной на оценке ресурсов узлов [10].

СПИСОК ЛИТЕРАТУРЫ

1. *Шальнев И. О.* Подход к построению распределенной виртуальной машины на основе объектно-ориентированного программирования // Изв. Тульского гос. ун-та. Технические науки. 2020. Вып. 9. С. 40—47.
2. *Smith J. E., Nair R.* The Architecture of Virtual Machines // Computer. 2005. Vol. 38, N 5. P. 32—38, 395–396. DOI: 10.1109/MC.2005.173.
3. *Турилин И. И., Галалу В. Г., Дагаев А. В.* Виртуальные машины, операционные системы и приложения. Таганрог, 2015. 64 с.
4. *Лафоре Р.* Объектно-ориентированное программирование в С++. СПб: Питер, 2016. 928 с.
5. *Радченко Г. И.* Распределенные вычислительные системы. Челябинск, 2012. 176 с.
6. *Thomborson C. D., Nicolescu R.* Test bed for Distributed Object Technologies using Java. The University of Auckland, Information Technology Faculty, 1999. 263 p.
7. *Lindholm T., Yellin F., Bracha G., Buckley A.* The Java Virtual Machine Specification. Oracle America, Inc., 2015. 604 p.
8. *Zaleski M.* YETI: a graduallY Extensible Trace Interpreter // Proc. of the 3rd Intern. Conf. on Virtual Execution Environments (VEE 2007). San Diego, California, USA, 13—15 June 2007. 155 p. DOI:10.1145/1254810.1254823.
9. *Шальнев И. О.* Подход к построению распределенных систем на основе балансировки объема исполняемого кода между сетевыми узлами // Матер. 4-й Междунар. науч. конф. „Технологическая перспектива: новые рынки и точки экономического роста“. СПб: Центр науч.-информ. техн. „Астерион“, 2018. С. 151—158.
10. *Шальнев И. О.* Объектно-ориентированный подход к описанию взаимодействия группы робототехнических средств // Изв. ЮФУ. Технические науки. 2021. № 1. С. 125—137. DOI 10.18522/2311-3103-2021-1-125-137.

Сведения об авторах

Сергей Викторович Кулешов — д-р техн. наук, профессор РАН; Санкт-Петербургский федеральный исследовательский центр Российской академии наук, лаборатория автоматизации научных исследований, Санкт-Петербургский институт информатики и автоматизации Российской академии наук; главный научный сотрудник; E-mail: kuleshov@iias.spb.su

Илья Олегович Шальнев — Санкт-Петербургский федеральный исследовательский центр Российской академии наук, лаборатория автоматизации научных исследований, Санкт-Петербургский институт информатики и автоматизации Российской академии наук; младший научный сотрудник; E-mail: shalnev.i@iias.spb.su

Поступила в редакцию 29.08.22; одобрена после рецензирования 14.09.22; принята к публикации 27.12.22.

REFERENCES

1. Shalnev I.O. *Izvestiya Tula State University (Izvestiya TulGU)*, 2020, no. 9, pp. 40–47. (in Russ.)
2. Smith J.E., Nair R. *Computer*, 2005, no. 5(38), pp. 32–38, 395–396, DOI:10.1109/MC.2005.173.
3. Turilin I.I., Galalu V.G., Dagaev A.V. *Virtual'nyye mashiny, operatsionnyye sistemy i prilozheniya* (Virtual Machines, Operating Systems and Applications), Taganrog, 2015, 64 p. (in Russ.)
4. Lafore R. *Object-Oriented Programming in C++*, Sams Publishing, 2002.
5. Radchenko G.I. *Raspredelennyye vychislitel'nyye sistemy* (Distributed Computing Systems), Chelyabinsk, 2012, 176 p. (in Russ.)
6. Thomborson C.D., Nicolescu R. *Test bed for Distributed Object Technologies using Java*, The University of Auckland, Information Technology Faculty, 1999, 263 p.
7. Lindholm T., Yellin F., Bracha G., Buckley A. *The Java Virtual Machine Specification*, Oracle America, Inc., 1997, 2015, 604 p.
8. Zaleski M. *Proceedings of the 3rd International Conference on Virtual Execution Environments, VEE 2007*, San Diego, California, USA, June 13–15, 2007, 155 p., DOI:10.1145/1254810.1254823.
9. Shalnev I.O. *Tekhnologicheskaya perspektiva: novyye rynki i tochki ekonomicheskogo rosta* (Technological Perspective: New Markets and Points of Economic Growth), Proceedings of the 4th International Scientific Conference, St. Petersburg, 2018, pp. 151–158. (in Russ.)
10. Shalnev I.O. *Izvestiya SFEDU. Engineering Sciences*, 2021, no. 1, pp.125–137, DOI 10.18522/2311-3103-2021-1-125-137. (in Russ.)

Data on authors

Sergey V. Kuleshov — Dr. Sci., Professor of the RAS; St. Petersburg Federal Research Center of the RAS, St. Petersburg Institute for Informatics and Automation, Research Automation Laboratory; Chief Researcher; E-mail: kuleshov@iias.spb.su

Ilya O. Shalnev — St. Petersburg Federal Research Center of the RAS, St. Petersburg Institute for Informatics and Automation, Research Automation Laboratory; Junior Researcher; E-mail: shalnev.i@iias.spb.su

Received 29.08.22; approved after reviewing 14.09.22; accepted for publication 27.12.22.