УДК 004.51

ПРОИЗВОДИТЕЛЬНОСТЬ ПРОГРАММНЫХ ИНТЕРФЕЙСОВ ВЗАИМОДЕЙСТВИЯ ВИРТУАЛЬНЫХ МАШИН JAVA И .NET

Ф. Ф. Иванов, А. В. Никифоров
Сургутский государственный университет, iff777@yandex.ru, klgdante@gmail.com.

В статье рассматривается вопрос производительности программных интерфейсов взаимодействия виртуальных машин Java и .NET. Тема производительности программных интерфейсов взаимодействия виртуальных машин Java и .NET раскрывается посредством описания хода эксперимента по определению наиболее производительного интерфейса из существующих. По результатам эксперимента делается вывод не только о производительности программных интерфейсов взаимодействия виртуальных машин Java и .NET, но также о подходе к проверке производительности для любых программных интерфейсов взаимодействия и ее обоснованности.

Ключевые слова: производительность программных интерфейсов взаимодействия, виртуальные машины, интерфейсы взаимодействия, программные интерфейсы, Java, .NET.

PERFORMANCE OF INTERACTION PROGRAM INTERFACES OF JAVA AND .NET VIRTUAL MACHINES

F. F. Ivanov, A. V. Nikiforov
Surgut State University, iff777@yandex.ru, klgdante@gmail.com.

The article discusses the performance of the Java and .NET virtual machine interaction program interfaces. The performance subject of the interaction program interfaces (IPI) of the Java and .NET virtual machines is revealed by describing the course of an experiment in determining the most productive interface from the existing ones. According to the results of the experiment, a conclusion is drawn not only about the performance of the program interfaces of interaction between the Java and .NET virtual machines, but also about the approach and the sense of testing performance for any program interfaces of interaction.

Keywords: performance of interaction program interfaces, virtual machines, interaction interfaces, program interfaces, Java, .NET.

На сегодняшний день при разработке программного обеспечения или информационных систем уделяется большое внимание производительности конечного продукта. Чем выше производительность программных интерфейсов взаимодействия (далее – ПИВ), тем быстрее программное обеспечение или информационная система решают поставленные перед ней задачи. Вопрос производительности не обходят стороной и программные интерфейсы взаимодействия, которые являются частью любого программного обеспечения или информационной системы.

Рассмотрим тему производительности программных интерфейсов взаимодействия на примере виртуальных машин Java [1] и .NET [2]. Для виртуальных машин Java и .NET существует два вида ПИВ:

1) сетевой интерфейс;
2) потоковый интерфейс.

Все подробности о приведенных видах программных интерфейсов взаимодействия и принципах их работы описаны в предыдущей статье [3].

Чтобы определить, какой из интерфейсов обладает наибольшей производительностью, был проведен эксперимент. Для проведения эксперимента были созданы две модели, кото-

рые выполняли расчет заданного количества членов арифметической и геометрической прогрессии. В каждой из моделей был реализован один из видов ПИВ – сетевой или потоковый. В качестве родительской виртуальной машины была выбрана виртуальная машина .NET, а в качестве дочерней – виртуальная машина Java. Данный аспект не играет существенной роли в ходе проведения эксперимента, поскольку возможна и обратная ситуация. Следующим шагом в проведении эксперимента являлся запуск моделей по 1 000 раз для сбора статистических данных. Измерение времени выполнения каждой из моделей для обоих интерфейсов проводилось при помощи бенчмарка «BenchmarkDotNet» [4, 5]. После завершения данного шага были получены две выборки суммарным объемом 2 000 значений. Далее была произведена предварительная обработка полученных в результате эксперимента значений. Из всех значений в каждой выборке были выделены уникальные значения времени выполнения и подсчитаны показатели частоты их повторения в выборке. Полученный результат для каждого из видов ПИВ представлен в табл. 1 и 2.

Таблица 1

Значения случайной величины времени выполнения потокового интерфейса

x_i (ms) – миллисекунды	f_i – частота
51,3	1
51,4	1
51,5	2
51,7	2
51,8	4
52,1	5
52,2	6
52,3	7
52,4	3
52,5	9
52,6	7
52,7	11
52,8	19
52,9	24
53	43
53,1	49
53,2	42
53,3	46
53,4	34
53,5	47
53,6	46
53,7	36
53,8	43
53,9	45
54	47
54,1	31
54,2	44
54,3	37
54,4	34
54,5	25
54,6	33
54,7	26
54,8	36
54,9	24
55	19
55,1	13
55,2	18
55,3	18

Окончание табл. 1

x_i (ms) – миллисекунды	f_i – частота
55,4	19
55,5	11
55,6	5
55,7	5
55,8	4
55,9	2
56	2
56,1	6
56,2	4
56,3	2
56,6	2
77,5	1

Таблица 2

Значения случайной величины времени выполнения сетевого интерфейса

x_i (ms) – миллисекунды	f_i – частота
69,3	1
70,6	1
71,5	1
71,9	2
72	1
72,1	1
73	1
73,4	1
73,6	2
74	1
74,1	1
74,2	1
74,3	1
74,4	2
74,5	1
74,6	3
74,7	2
74,8	43
74,9	81
75	93
75,1	97
75,2	102
75,3	134
75,4	114
75,5	84
75,6	92
75,7	52
75,8	32
75,9	22
76	9
76,1	3
76,2	2
76,3	3
76,4	1
76,6	2
76,8	1
77,3	1
77,4	2
77,5	2

Окончание табл. 2

x_i (ms) – миллисекунды	f_i – частота
77,6	1
77.8	2
78,8	1
82,7	1

Примечание: x_i – значение случайной величины времени выполнения модели в ms, а f_i – частота появления x_i случайной величины времени выполнения в выборке.

Для определения производительности ПИВ использовался аппарат математической статистики. Этого будет достаточно для получения сравнительной оценки производительности ПИВ [6].

В табл. 3 представлены результаты расчетов среднего значения выборки и стандартного отклонения для обоих видов интерфейсов, а в табл. 4 содержатся интервалы, полученные по правилу 3-х сигм. Также на рис. 1 и 2 показаны графики для каждого вида интерфейсов. На каждом графике присутствуют все уникальные значения и подсчитана их частота для каждого из интерфейсов, а также отмечены интервалы, полученные по правилу 3-х сигм.

Таблица 3

Расчетные значения среднего и стандартного отклонения выборок

	Потоковый интерфейс	**Сетевой интерфейс**
$\bar{x}$ **(ms)**	53,96	75,3
σ (ms)	1,15	0,58
$\lvert\overline{x_п} - \overline{x_с}\rvert$	21,34	
$\lvert\overline{\sigma_п} - \overline{\sigma_с}\rvert$	0,56	

Примечание: $\bar{x}$ – среднее значение случайной величины времени выполнения модели, σ – стандартное отклонение случайной величины времени выполнения, а $\lvert\overline{x_п} - \overline{x_с}\rvert$ *и* $\lvert\overline{\sigma_п} - \overline{\sigma_с}\rvert$ – разность по модулю среднего значения и стандартного отклонения между обоими интерфейсами.

Таблица 4

Расчетные значения интервалов

	Потоковый интерфейс	**Сетевой интерфейс**
[$\bar{x}$ - σ ; $\bar{x}$ + σ]	[52,81; 55,1]	[74,71; 75,88]
[$\bar{x}$ - 2σ ; $\bar{x}$ + 2σ]	[51,67; 56,25]	[74,13; 76,46]
[$\bar{x}$ - 3σ ; $\bar{x}$ + 3σ]	[50,52; 57,4]	[73,55; 77,05]

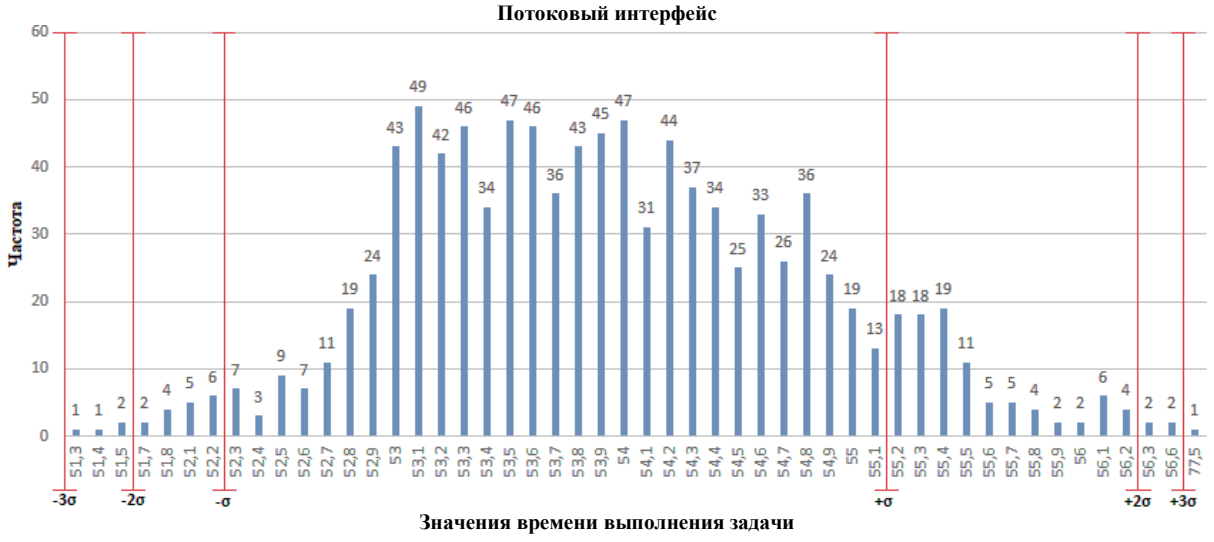

Рис. 1. Гистограмма частот времени выполнения задачи при потоковом интерфейсе

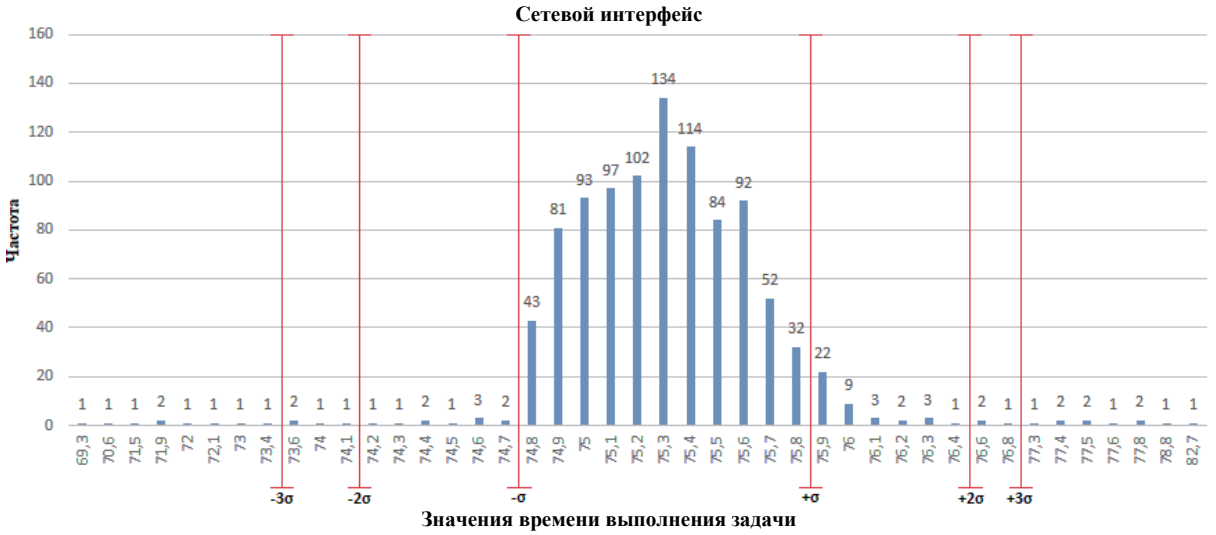

Рис. 2. Гистограмма частот времени выполнения задачи при сетевом интерфейсе

По графикам, представленным на рис. 1 и 2, видно распределение значений времени выполнения модели по интервалам. По табл. 5 видно, какое количество значений времени выполнения равно среднему, лежит в одном из трех интервалов или не попадает ни в один из интервалов.

Таблица 5

Количество случайных величин времени выполнения, равных среднему и входящих в интервалы

	Потоковый интерфейс	**Сетевой интерфейс**
$[\bar{x} - \sigma ; \bar{x} + \sigma]$	824	924
$[\bar{x} - 2\sigma ; \bar{x} + 2\sigma]$	991	974
$[\bar{x} - 3\sigma ; \bar{x} + 3\sigma]$	999	981
$(-\infty ; \bar{x} - 3\sigma)$	0	9
$(\bar{x} + 3\sigma ; +\infty)$	1	10

По данным из табл. 3 и 5 можно сделать следующие выводы по производительности ПИВ виртуальных машин Java и .NET.

Из данных в табл. 5 видно, что количество вхождений значений случайной величины времени выполнения на интервале $[\bar{x} - \sigma; \bar{x} + \sigma]$ у сетевого интерфейса выше, чем у потокового. Однако на интервалах $[\bar{x} - 2\sigma; \bar{x} + 2\sigma]$ и $[\bar{x} - 3\sigma; \bar{x} + 3\sigma]$ у потокового интерфейса наблюдается большее количество вхождений значений случайной величины времени выполнения, чем у сетевого. Благодаря этому потоковый интерфейс имеет только одно значение, выходящее за интервал в 3 сигме, а сетевой интерфейс имеет 19 таких значений. И это, несмотря на то, что стандартное отклонение сетевого интерфейса меньше на 0,56 ms, чем у потокового. Данные факты говорят о том, что потоковый интерфейс, в отличие от сетевого, более устойчив перед внешними воздействиями.

Такая ситуация обусловлена тем, что потоковый интерфейс работает на одном компьютере. Сетевой интерфейс же работает на двух компьютерах и требует сетевого соединение между ними. Именно сетевое соединение и второй компьютер оказывают влияние на устойчивость интерфейса перед внешними воздействиями.

Также при сравнении средних значений времени выполнения потоковый интерфейс выполняется на 21,34 ms быстрее, чем сетевой. Что и дает прирост в производительности ≈ 40 % при использовании потокового интерфейса в сравнении с сетевым. Однако такой прирост производительности потокового интерфейса справедлив только для моделируемой в ходе эксперимен-

та задачи по расчету членов арифметической и геометрической прогрессии. Для другого типа задач прирост производительности потокового интерфейса может уменьшиться или увеличиться в сравнении с модельной задачей. Например, прирост производительности потокового интерфейса для задачи конвертации XML в PDF составил всего ≈ 2 % [7, 8].

Из всего вышесказанного можно сделать вывод, что потоковый интерфейс обладает большей устойчивостью к внешним воздействиям, т. е. является более стабильным при работе в сравнении с сетевым. Также общее время выполнения задачи меньше при использовании потокового интерфейса, а не сетевого. Поэтому потоковый интерфейс для большинства выполняемых задач на виртуальных машинах Java и .NET является наиболее производительным программным интерфейсом взаимодействия. Однако некоторые типы задач могут быть исключением.

Поэтому, несмотря на полученные в ходе эксперимента результаты, хотелось бы отметить, что производительность любых ПИВ зависит от условий внешней среды и задач, для которых они применяются. Это относится и к программным интерфейсам взаимодействия Java и .NET. Правильным решением будет проверка производительности интерфейсов на конкретных задачах и в конкретных условиях внешней среды. Это позволит выбрать оптимальный в плане производительности программный интерфейс взаимодействия для разрабатываемого программного обеспечения или информационной системы.

Литература

1. Виртуальная машина Java. URL: https://ru.wikipedia.org (дата обращения: 10.02.2019).

2. Виртуальная машина .NET. URL: https://ru.wikipedia.org (дата обращения: 10.02.2019).

3. Программные интерфейсы взаимодействия виртуальных машин Java и .NET. URL: http://www.surgu.ru/attachment/16982/download/Sbornik%20materialov%20HHII%20Otkrytoy%20regionalnoy%20studencheskoy%20nauchnoy%20konferentsii%20im-%20G-%20I-%20Nazina%20%C2%ABNAUKA%2060-y%20PARALLELI%C2%BB,%2004%20aprelya% 202018%20g-.pdf (дата обращения: 10.02.2019).

4. Бенчмарк. URL: https://dic.academic.ru (дата обращения: 10.02.2019).

5. BenchmarkDotNet. URL: https://benchmarkdotnet.org (дата обращения: 10.02.2019).

6. Гмурман В. Е. Теория вероятностей и математическая статистика. М. : Юрайт, 2018. с. 479.

7. Интеграция виртуальных машин .NET и Java : докл. URL: https://youtu.be. (дата обращения: 10.02.2019).

8. Интеграция виртуальных машин .NET и Java : презентация. URL: https://docplayer.ru/50861859-Integraciya-virtualnyh-mashin-net-i-java.html (дата обращения: 10.02.2019).