МЕТОДЫ ТЕСТИРОВАНИЯ ИНТЕГРИРОВАННЫХ ИНФОРМАЦИОННЫХ СИСТЕМ

Туровец Николай Олегович,
Белорусский государственный университет информатики и радиоэлектроники,
г. Минск, Республика Беларусь

E-mail: turovets.nick999@gmail.com

Алефиренко Виктор Михайлович,
Белорусский государственный университет информатики и радиоэлектроники,
г. Минск, Республика Беларусь

E-mail: alefirenko@bsuir.by

Аннотация. Статья посвящена обзору методов тестирования интегрированных информационных систем. Рассмотрены основные инструменты для проведения тестирования производительности, нагрузочного тестирования и стресс-тестирования.

Ключевые слова: информационная система, виды тестирования, тестирование производительности, нагрузочное тестирование, стресс-тестирование.

Интегрированная информационная система – это набор интегрированных приложений, которые комплексно, в едином информационном пространстве поддерживают все основные аспекты управленческой деятельности предприятия. В настоящее время существует большое количество различных интегрированных информационных систем [1]. Перед внедрением в эксплуатацию необходимо проводить тестирование информационной системы.

Работа любого программного обеспечения информационной системы должна предусматривать нагрузку в течение длительного времени. Сбои и отказы в системе могут привести к неприятным последствиям, таким как потеря пользователей и убытки компании. Тестирование позволяет определить, как и с какой скоростью может работать система под определенной нагрузкой. Можно выделить 3 основных метода тестирования информационных систем: тестирование производительности, нагрузочное тестирование и стресс-тестирование.

Тестирование производительности интегрированной информационной системы позволяет находить возможные уязвимости и недостатки в системе в

условиях нормальной, повышенной и пиковой нагрузки. С помощью нагрузочного тестирования проводится оценка на соответствие производительности продукта требованиям, сформулированным в техническом задании. Нагрузочное тестирование выполняется для определения времени отклика системы на запросы различных типов, с целью удостовериться, что система работает в соответствии с требованиями при нормальной пользовательской нагрузке. Стресс-тестирование предполагает проверку работоспособности системы при максимальных нагрузках и нагрузках, превышающих пользовательские в несколько раз.

Тестирование производительности (*Performance Testing*) – это тестирование, которое выполняется для определения того, как компоненты системы работают в конкретной ситуации [2].

В большинстве случаев тестирование производительности имеет не одну, а несколько целей, так как другие типы тестирования в ходе проведения совмещаются с другими целями. Основная цель тестирования производительности включает установление эталонного поведения системы. Основная особенность тестирования производительности заключается в том, что оно происходит только после полного функционального тестирования. Ошибки функциональности не исправляются в ходе тестирования производительности [2].

Тестирование производительности проводится с целью предоставления информации о системе, касающейся скорости, стабильности и масштабируемости. Тестирование производительности определяет, соответствует ли программное обеспечение информационной системы требованиям скорости, масштабируемости и стабильности при ожидаемых рабочих нагрузках [2].

В настоящее время на рынке программного обеспечения существует большое количество различных инструментов для проведения тестирования производительности, начиная от программных средств с открытым исходным кодом и заканчивая приложениями для автоматизированного тестирования премиум-класса. Однако, имея такое множество альтернатив, иногда трудно выбрать лучший инструмент. Рассмотрим несколько инструментов тестирования производительности [3].

Apache JMeter – один из самых популярных инструментов с открытым исходным кодом для тестирования производительности. Инструмент имеет богатую функциональность и достаточно давно успел себя зарекомендовать, как отличную альтернативу платным аналогам для большинства решаемых задач [3].

Инструмент кроссплатформенный, так как разработан на *Java*. Доступна как работа из *GUI*, так и запуски в консольном режиме. *JMeter* действительно очень популярен в отрасли, об этом говорит тот факт, что некоторые инструменты используют *JMeter* в качестве своего базового программного обеспечения. Интерфейс *JMeter* представлен на рисунке 1.

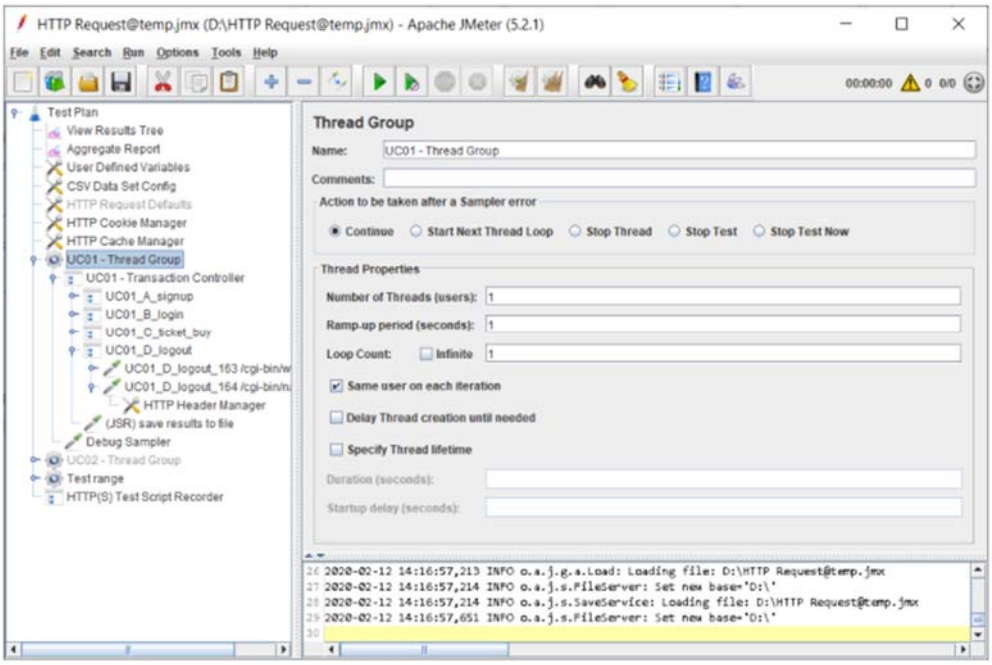

Рис. 1 Интерфейс *Apache JMeter*

Apache JMeter – это отличный инструмент тестирования производительности с открытым исходным кодом для больших и малых компаний. Программа предоставляет бесплатно своим пользователям различные полезные инструменты тестирования, и пользователи могут настроить её в соответствии со своими потребностями и каждого.

Micro-Focus LoadRunner (ранее известный как *HP LoadRunner*) – это довольно сложный и универсальный инструмент тестирования производительности программного обеспечения, который обнаруживает проблемы с производительностью прежде всего в корпоративных приложениях, используемых крупными компаниями для зарабатывания денег. Однако из-за своей цены этот инструмент больше подходит для средних и крупных организаций. *LoadRunner* может применяться для тестирования программного обеспечения *ERP*, устаревших системных приложений, а также технологий *Web* 2.0. Интерфейс *LoadRunner* представлен на рисунке 2 [3].

Рис. 2 Интерфейс *Micro-Focus Loadrunner*

LoadRunner позволяет тестировщикам программного обеспечения проводить комплексную оценку производительности своей системы. Он специализируется на выявлении узких мест до того, как приложение будет внедрено в компанию или до стадии развертывания. В результате пользователи могут оценить каждый компонент по отдельности, прежде чем он начнет работать. Также этот инструмент очень полезен при обнаружении пробелов в производительности, если будет проводится обновление системы. Также он предоставляет пользователям функции для прогнозирования затрат по увеличению производительности приложений. Благодаря точному прогнозированию таких затрат, связанных с аппаратным и программным обеспечением, специалистам проще повысить производительность и масштабируемость приложения [3].

BlazeMeter – инструмент тестирования, который предоставляет пользователям тестирование производительности, нагрузочное тестирование и стресс-тестирование как услугу. Служба содержит инновационную и всеобъемлющую платформу непрерывного тестирования. Веб-интерфейс приложения эффективен для создания статических нагрузочных тестов и использования сценариев *JMeter* для выполнения динамических нагрузочных тестов. Интерфейс *BlazeMeter* представлен на рисунке 3 [3].

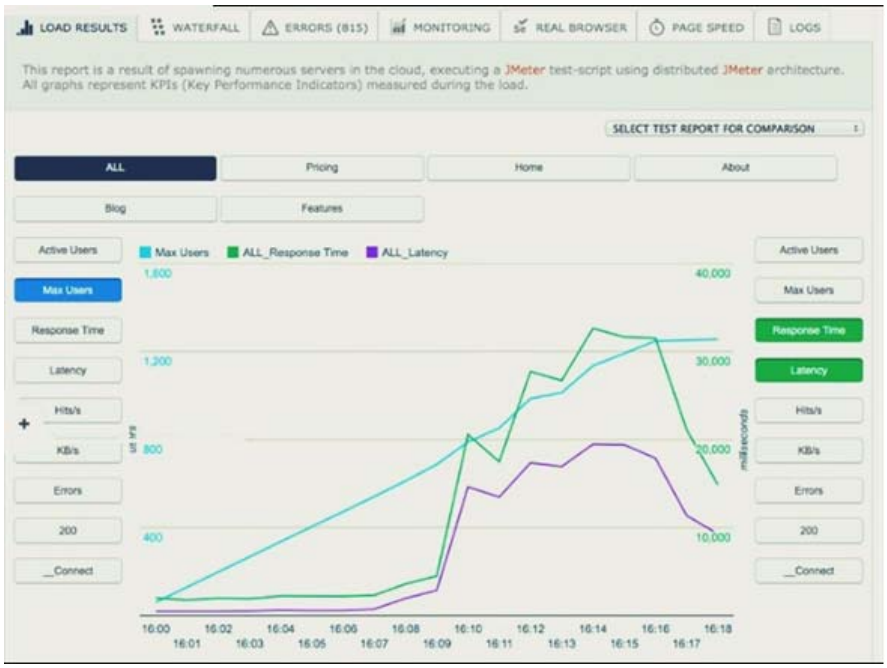

Рис. 3 Интерфейс *BlazeMeter*

BlazeMeter известен широчайшим использованием одного из лучших инструментов нагрузочного тестирования с открытым исходным кодом – *Apache JMeter*. Он предоставляет различные корпоративные функции для бесплатной платформы. То есть пользователи могут получить доступ ко многим расширенным функциям, таким как мониторинг производительности приложений (APM), создание отчетов в режиме реального времени, распределенное тестирование и интеграция с инструментами разработчика для непрерывной интеграции [3].

BlazeMeter – отличный инструмент для тестирования производительности для организаций, которые уже используют *Apache JMeter*.

Тестирование производительности рекомендуется проводить при внедрении крупного изменения, которое существенно влияет на функционал (например, обновление программного обеспечения). Оно позволяет выявить и предотвратить ошибки в информационной системе перед внедрением в эксплуатацию; определить максимальное количество одновременно работающих пользователей в системе и максимальное количество одновременно выполняемых операций без потери качества обслуживания.

Нагрузочное тестирование (*load testing*) – данный тип тестирования представляет собой определение производительности и времени-отклика программно-технической системы в ответ на внешний запрос, а также позволяет оценить поведение системы при возрастающей нагрузке. Цель нагрузочного тестирования – это определение максимальной нагрузки, при которой система может функционировать.

В роли нагрузки может выступать количество пользователей, а также количество операций на сервере.

Производительность при этом определяется следующими факторами [4]:

– скоростью работы программного обеспечения;

– скоростью работы аппаратного обеспечения;

– скоростью работы сети.

Во время тестирования могут осуществляться следующие операции, позволяющие более точно измерить производительность и определить «узкое место» системы [4]:

– измерение времени выполнения выбранных операций при определенных интенсивностях выполнения этих операций;

– определение количества пользователей, одновременно работающих с приложением;

– определение границ приемлемой производительности при увеличении нагрузки (при увеличении интенсивности выполнения этих операций).

Рассмотрим 2 примера тестов, которые представлены на рисунке 4 и 5, для определения количества пользователей, с которыми может работать система.

Пример 1. В этом тесте 50 пользователей добавляются через каждые 50 секунд, пока нагрузка не достигнет 500 пользователей. Каждый шаг занимает 50 секунд, и *JMeter* ждет 50 секунд, прежде чем начать следующий шаг. Как только нагрузка достигнет 500 потоков, все они будут продолжать работать в течение 100 секунд вместе, а затем, наконец, останавливают 20 потоков каждые 10 секунд [5].

Пример 2. В этом тесте 10 пользователей добавляются через каждые 120 секунд, пока нагрузка не достигнет 100 пользователей. Каждый шаг занимает 120 секунд, и *JMeter* ждет 120 секунд, прежде чем начать следующий шаг. Как только нагрузка достигнет 100 потоков, все они будут продолжать работать в течение 180 секунд вместе, а затем, наконец, останавливают 5 потоков каждые 10 секунд [6].

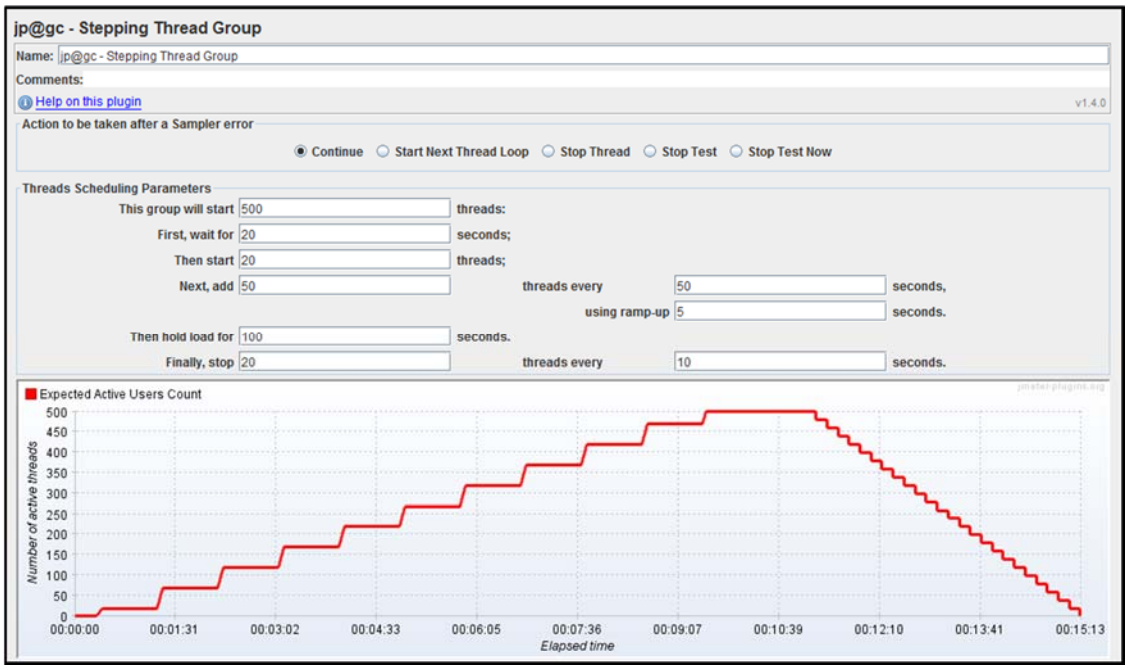

Рис. 4 Пример 1 выполнения нагрузочного тестирования

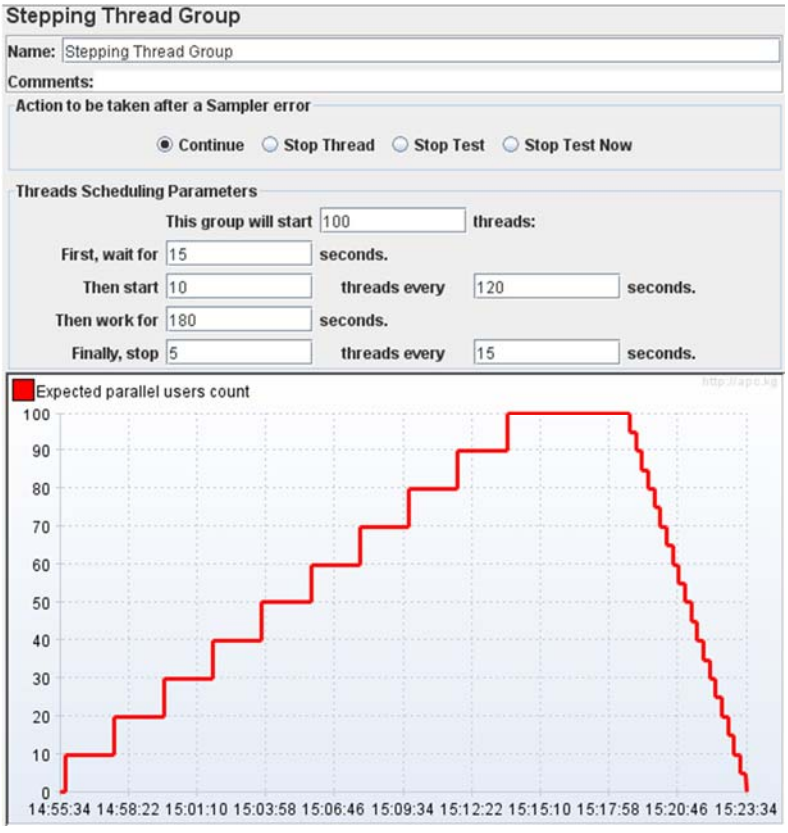

Рис. 5 Пример 2 выполнения нагрузочного тестирования

Стресс-тестирование (*stress-testing*) – это вид тестирования, который характеризует систему со стороны устойчивости её работы при условиях, превышающих нормальные. Цель данного вида тестирования – оценка производительности системы при пороговых значениях рабочей нагрузки или за её пределом. Также в ходе тестирования можно оценивать работу системы при изменении ресурсов, доступных системе, таких как процессорное время, память, ширина сетевого канала и т. д [7].

В ходе тестирования измеряется [2]:

– возможность и время регенерации системы – возможность и время возвращения системы к нормальному состоянию после стрессовых нагрузок;

– корректность логирования ошибок и оповещений о их возникновении;

– производительность системы при стрессовой нагрузке;

– оценка влияния сбоев тестируемой системы на внешние системы.

Проверять работоспособность в экстремальных условиях использования в первую очередь необходимо для программного обеспечения, сбои в работе которого может привести к большим и серьезным последствиям, как финансовым, так и более серьезных, вплоть до человеческих жизней [2].

Зачастую, вручную создать необходимые условия нагрузки не представляется возможным, поэтому прибегают к [8]:

– фокусировании на транзакциях определенных типов, что более сильно влияет на возникновение граничных ситуаций, нежели при нагрузочном тестировании;

– использованию скриптов, виртуальных пользователей и прочих автоматизированных симуляторов интенсивного рабочего процесса, что позволяет исследовать поведение программного обеспечения при пиковой нагрузке;

– исследованию «узких мест» системы;

– исследованию процессов обработки ошибок и исключительных ситуаций.

Таким образом, в настоящее время существует большое число инструментов тестирования информационных систем. Представленные методы и соответствующие инструменты позволяют проверить такие нефункциональные требования, как производительность и работоспособность информационной системы при различных нагрузках и создаваемых условиях. Тестирование позволяет не только следить за работой системы, но и помогает давать рекомендации по оптимизации работы системы, а также позволяет выявить и устранить большой набор потенциальных угроз, которые информационная система может начать испытывать на стадии эксплуатации.

Литература:

1. Алефиренко В.М. Обзор и классификация информационных систем / В.М. Алефиренко, Н.О. Туровец // Danish Scientific Journal. – 2021. – Vol. 1, № 55. – С. 52-56.
2. PerfomanceLab [Электронный ресурс]. – Режим доступа: https://www.performance-lab.ru/blog/load-testing/testirovanie-proizvoditelnosti
3. PerfomanceLab [Электронный ресурс]. – Режим доступа: https://www.performance-lab.ru/blog/luchshie-instrumenty-dlya-nagruzochnogo-testirovaniya
4. ПроТестинг.ru [Электронный ресурс]. – Режим доступа: http://www.protesting.ru/testing/types/loadtesttypes.html
5. BlazeMeter [Электронный ресурс]. – Режим доступа: https://www.blazemeter.com/blog/advanced-load-testing-scenarios-jmeter-part-4-stepping-thread-group-and-concurrency-thread
6. Jmeter-Plugins [Электронный ресурс]. – Режим доступа: https://jmeter-plugins.org/wiki/SteppingThreadGroup/
7. QALight [Электронный ресурс]. – Режим доступа: https://qalight.ua/ru/baza-znaniy/stress-testirovanie/
8. Software-Testing [Электронный ресурс]. – Режим доступа: https://software-testing.ru/library/testing/performance-testing