АНАЛИЗ СРЕДСТВ РАЗРАБОТКИ И ИНСТРУМЕНТОВ АВТОМАТИЗАЦИИ НАГРУЗОЧНОГО ТЕСТИРОВАНИЯ

Сорокина Татьяна Александровна

студент, Московский государственный технологический университет «СТАНКИН», РФ, г. Москва

В процессе разработки нагрузочных тестов используются следующие инструменты и средства автоматизации: среда разработки, программные решения для разработки заглушек, система для формирования графических представлений по мониторингам различных метрик. Также определяется язык программирования.

Выбор языка программирования для разработки нагрузочных авто-тестов зависит от нескольких факторов:

Требования к производительности и масштабируемости – некоторые языки программирования обладают низкой производительностью, данный критерий может повлиять на скорость выполнения тестов, что является неприемлемым при проведении нагрузочного тестирования.

Язык программирования должен поддерживать различные плагины и библиотеки, данная возможность может значительно упростить процесс создания и выполнения авто-тестов.

Совместимость с используемыми технологиями – необходимо учитывать, что выбранный язык программирования должен быть совместим с технологиями, используемыми в проекте, чтобы обеспечить эффективную работу нагрузочных тестов.

Основными языками программирования для написания тестов являются «Java» и «Groovy».

«Java» – один из самых распространенных объектно-ориентированных языков, используемых при разработке авто-тестов. Данный язык программирования обладает следующими преимуществами в нагрузочном тестировании:

1. Обладает высокой производительностью – скорость обработки фрагментов когда в нагрузочном тестировании является важным фактором.
2. Обладает эффективными сетевыми технологиями – имеется обширная программная библиотека для передачи данных по распространённым протоколам: «FTP», «HTTP», «TCP/IP», также работает механизм вызова удалённых методов. Данный критерий является значимым, так как при разработке тестовых скрипов используются «HTTP» запросы, которые имитируют вызов бэка фронтовыми компонентами.
3. Язык поддерживает обширное количество фреймворков, что позволит значительно сократить объем программного кода.

Недостатками языка являются:

1. Значительное использование памяти – управление памятью в «Java» может быть неэффективным, что негативно сказывается на производительности.

2. Сложность языка – «Java» использует большое количество сложных синтаксических конструкций, которые могут быть трудными для запоминания и чтения.

Язык «Groovy» – интерпретируемый язык программирования с объектно-ориентированной

парадигмой и «Java»-подобным синтаксисом. Выделим основные преимущества языка:

1. Главное преимущество использования «Groovy» заключается в его возможности уменьшить количество кода, необходимого для выполнения определенной задачи. Он предлагает более высокий уровень абстракции, что позволяет программистам писать более компактный код.

2. Имеется программная библиотека для передачи данных по протоколам «HTTP», «TCP/IP».

Недостатками языка являются:

1. Необходимость использования виртуальной машины, что повышает стоимость разработки.

2. Низкая производительность.

3. Отсутствие средства форматирования исходного кода.

Изучив основные преимущества и недостатки языков, можно сделать выбор в пользу «Java». Данный язык соответствует требованиям к производительности, совместим с технологиями проекта и обладает обширным количеством фреймворков.

Выбор среды разработки базируется на следующих критериях:

Удобство использования: среда разработки должна быть доступной для понимания без длительного изучения инструкций и обладать удобными инструментами для создания и настройки нагрузочных тестов.

Функциональность: среда разработки должна обладать всем необходимым функционалом для создания и выполнения нагрузочных тестов – необходима возможность добавления тестовых сценариев, настройки нагрузки и мониторинга, возможность генерации уникальных переменных.

Поддержка технологий: среда должна поддерживать необходимые технологии и протоколы для тестирования системы. Например, некоторые среды разработки могут поддерживать только «HTTP», в то время как другие могут поддерживать различные протоколы, такие как «WebSocket» или «MQTT».

Производительность и масштабируемость: среда разработки должна обеспечивать достаточную производительность для выполнения нагрузочных тестов на больших нагрузках и быть масштабируемой для работы с различными типами приложений и сервисов.

Совместимость с другими инструментами: выбор среды разработки также зависит от того, с какими сторонними инструментами она совместима. Например, если разработчики уже используют определенные инструменты для автоматизации тестирования, то им может быть удобнее выбрать среду разработки, которая интегрируется с этими инструментами. Важным требованием является совместимость с системой «Kafka», так как она является неотъемлемым инструментом тестируемой системы.

Также важным критерием при выборе среды разработки является поддержка выбранного языка программирования.

В качестве сред разработки можно выделить следующие системы:

«Jmeter» – инструмент разработки и проведения нагрузочного тестирования, представляющий собой приложение с открытым исходным кодом на базе «Java» [1].

Инструмент обладает следующими преимуществами:

1. Имеет обширный функционал для создания и выполнения нагрузочных тестов – составление сценариев тестирования, алгоритмы для генерации данных, механизмы авторизации виртуальных пользователей, логирование результатов теста.

2. Поддерживает технологии «SOAP», «HTTP» и «TCP».

3. Обладает высокой производительностью.

4. Архитектура системы позволяет поддерживать плагины сторонних разработчиков, также поддерживает интеграцию с «Kafka».

5. Поддерживает «Java» код.

Основными недостатками данного инструмента являются:

1. Сложный интерфейс – для новых пользователей может быть сложной задачей понять всю функциональность, необходимо подробное изучение документации.

2. Системные ограничения и аппаратные возможности – «JMeter» требует установки на локальной машине. Это может быть трудоёмко и требует дополнительных инвестиций в оборудование.

«Gatling» - это платформа тестирования нагрузки и производительности с открытым исходным кодом.

Преимуществами данной системы являются:

1. Лаконичный и понятный пользовательский интерфейс с функциональными подсказками и поиском функциональных составляющих.

2. Высокая скорость генерации запросов.

3. Возможность моделирования пользовательских сценариев.

4. Поддерживает технологии «HTTP» и «HTTPS».

5. Архитектура системы позволяет поддерживать интеграцию с «Kafka».

6. Поддерживает «Java» код.

Недостатками данной системы являются:

1. Использует много ресурсов системы при генерации нагрузки.

2. Имеет ограниченные возможности сценариев.

3. Не все версии поддерживают интеграцию с «Kafka».

Изучив основные преимущества и недостатки сред разработки, можно сделать выбор в пользу «Jmeter». Данная система в большей степени соответствует критериям выбора, также «Gatling» обладает важными минусами: использование большого количества ресурсов системы может повлиять на качество проведения нагрузочного тестирования, в виду сложности тестируемой системы часть пользовательских сценариев автоматизировать не получится.

Выбор программных решений для разработки мок заглушек базируется на следующих критериях:

Функциональность – программа должна обладать достаточным набором возможностей для создания качественных заглушек. Особыми критериями является наличие возможности формирования тела ответа в формате «Json» и имитация сбоев микросервисов и базы данных.

Удобство использования – интерфейс программы должен быть интуитивно понятным и удобным для работы. Данный критерий обусловлен относительной простотой заглушки, если

интерфейс будет сложным, времени на изучение системы уйдет больше, чем на разработку заглушек.

Поддерживаемые форматы файлов – программа должна поддерживать широкий спектр форматов файлов, чтобы обеспечить гибкость в работе.

Совместимость – программа должна быть совместима с другими программными решениями, в данной работе система должна быть совместима с «OpenShift».

Рассмотрим следующие системы:

«WireMock» – это библиотека на «java» для создания «HTTP» заглушек над веб-сервисами. Программное решение позволяет создать «HTTP» -сервер, к которому программный модуль может подключиться, как к реальному веб-сервису [2].

Основными преимуществами системы являются:

1. Функциональные возможности системы обширны, позволяют имитировать сбои компонентов, поддерживается формат «Json», ответы учитывают особенности входящего запроса (обращения) к заглушке.

2. Не имеет интерфеса, так как является библиотекой, при разработке простых заглушек, обращение к документации не требуется.

3. Поддерживает форматы «HTML», «XML», «JSON».

4. Совместим с «OpenShift».

Основным недостатком системы: для вызова заглушек через «OpenShift» необходимо настраивать интеграцию, на данную манипуляцию уходит значительное количество времени.

Рассмотрим приложение «SoapUI».

«SoapUI» – это приложение с открытым исходным кодом для тестирования веб-сервисов сервис-ориентированных архитектур (SOA) и передачи состояний представлений (REST) [3].

Основными преимуществами приложения являются:

1. Позволяет имитировать сбои компонентов, ответы учитывают особенности входящего запроса (обращения) к заглушке.

2. Поддерживает форматы «HTML», «XML», «JSON».

3. Совместим с «OpenShift», для реализации интеграции не требуется большого количества времени.

Основными недостатками приложения являются:

1. Сложный интерфейс с минимальной документацией.

2. Необходимо использовать «VPN» при работе с приложением.

Изучив основные преимущества и недостатки систем для разработки мок заглушек, можно сделать выбор в пользу «WireMock».

Для формирования графических представлений рассмотрим самые востребованные системы – «Grafana» и «Kibana».

Критериями выбора систем для формирования графических представлений по результатам мониторинга являются:

Возможности настройки и кастомизации – система должна предоставлять возможность настройки графических представлений в соответствии с потребностями и требованиями пользователя.

Поддержка различных типов графиков – выбранная система должна поддерживать различные типы графиков (линейные, столбчатые, круговые и т. д.) для создания разнообразных представлений данных.

Возможность анализа данных – система должна обеспечивать возможность проведения анализа данных на основе представленных графиков для выявления тенденций и паттернов.

Интеграция с другими инструментами – система должна поддерживать интеграцию с другими инструментами и платформами для обмена данными и улучшения процесса мониторинга.

Стоимость и лицензионная политика – необходимо учитывать стоимость использования выбранной системы формирования графических представлений и ее лицензионную политику для оптимизации расходов.

«Grafana» – это свободная программная система визуализации данных, ориентированная на данные систем мониторинга [4].

В процессе анализа данной системы выявлены следующие преимущества:

1. Имеет настраиваемые панели мониторинга для отображения данных из широкого спектра баз данных с использованием инструментов визуализации.

2. Поддерживает графические представления в виде тепловых карт, диаграмм, гистограмм.

3. Поддерживает интеграцию с «JMeter» и «Prometheus».

Недостатками системы являются:

1. Высокая стоимость.

2. Ограниченные настройки панели мониторинга относительно других систем.

3. Краткая и неполная документация.

«Kibana» - это панель визуализации данных с открытым исходным кодом. Используется для анализа временных рядов, журналов и мониторинга приложений.

В процессе изучения системы выявлены следующие преимущества:

1. Имеет настраиваемые панели мониторинга.

2. Бесплатный с использовании.

3. Поддерживает графические представления в виде диаграмм, карт и таблиц данных.

4. Поддерживает интеграцию с «JMeter».

Недостатками системы являются:

1. Небезопасная система.

2. Не поддерживает интеграцию с «Prometheus».

3. Не поддерживает непрерывный мониторинг показателей в режиме реального времени, необходимо анализировать журналы.

Изучив основные преимущества и недостатки систем мониторинга, можно сделать выбор в пользу «Grafana», так как данная система поддерживает интеграцию с «JMeter» и «Prometheus», а также является безопасной в использовании.

Список литературы:

1. Сайт «Хабр» [Электронный ресурс]. Режим доступа: https://habr.com/ru/companies/otus/articles/746504/, свободный (дата обращения 01.04.2024 г.).

2. Сайт «Хабр» [Электронный ресурс]. Режим доступа: https://habr.com/ru/companies/rostelecom/articles/679276/, свободный (дата обращения 02.04.2024 г.).

3. Сайт «Википедия» [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/SoapUI, свободный (дата обращения 02.04.2024 г.).

4. Сайт «Википедия» [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/Grafana, свободный (дата обращения 02.04.2024 г.).