РАЗРАБОТКА МИКРОСЕРВИСОВ НА ОСНОВЕ VERT.X

Рафеенко Е. Д., Кондратьева О. М., Соболева Т. В.

Белорусский государственный университет, Минск, Беларусь,
e-mail: rafeenko@bsu.by

В настоящее время распространен термин «микросервисная архитектура» как способ разработки приложений в виде набора независимо развертываемых сервисов. Приложение состоит из набора небольших сервисов, каждый из которых работает в собственном процессе и взаимодействует с остальными с помощью легковесных механизмов, как правило HTTP. Используя совокупность разбитых на мелкие гранулы архитектур микросервисов, можно ускорить поставку программного обеспечения и внедрить в практику самые новые технологии.

Такого рода сервисы построены на основе бизнес-потребностей и развертываются независимо с использованием полностью автоматизированной среды. Сами по себе эти сервисы могут быть написаны на разных языках и использовать разные технологии хранения данных.

Корпоративные приложения часто включают три основные части: пользовательский интерфейс (состоящий, как правило, из HTML страниц и javascript кода), база данных (как правило, реляционная) и сервер. Монолитный сервер — довольно очевидный способ построения подобных систем. Вся логика по обработке запросов выполняется в единственном процессе, само приложение разделено на классы, функции и namespaces.

Архитектура микросервисов использует библиотеки, но основной способ разбиения приложения — деление его на сервисы. Сервисы — это компоненты, выполняемые в отдельном процессе и взаимодействующие между собой через веб-запросы или remote procedure call.

Eclipse Vert.x – это проект с открытым исходным кодом от Eclipse Foundation, который предоставляет удобную модель для построения микросервисов [1]. Vert.x имеет API для разработки асинхронных сетевых приложений, и можно подобрать необходимые модули для приложения: взаимодействия с БД, мониторинга, аутентификации, логгирования, обнаружения сервисов и т.д. Vert.x – платформа полиглот, поддерживает ряд языков, использующих JVM, возможна разработка частей приложения на различных JVM языках.

Основные понятия, которые использует Vert.x – это verticle (вертикаль) и event bus (шина событий), позволяющая вертикалям взаимодействовать между собой.

Вертикаль – единица развертывания в Vert.x, она обрабатывает входящие события, приходящие в event loop, такие как получение входящих сетевых буферов или сообщений от других вертикалей. Vert.x API неблокирующее, поэтому важные концепции Vert.x следующие [2]:

- каждое событие должно быть обработано за *приемлемое* время и не блокировать поток – event loop;
- блокирующие операции *не должны* выполняться в потоке, связанном с event loop.

Вертикаль всегда выполняется в одном потоке, и нет необходимости заботиться о проблемах многопоточности внутри вертикали. Vert.x использует все CPU (или ядра CPU) компьютера путем создания потока на CPU. Один поток может отправлять сообщения многим вертикалям.

Входящие сетевые данные, которые получает поток – event loop, передаются в качестве событий соответствующим вертикалям. Вертикаль может быть развернута

несколько раз, она может развернуть другую вертикаль. Если вертикаль развернута более чем один раз, то события распределяются между экземплярами вертикали в порядке круговой очереди. Обработка события заключается в выполнении соответствующего обработчика (listener object), зарегистрированного вертикалью. После этого поток может отправлять следующее сообщение другой вертикали.

Event bus – связующее звено Vert.x приложения, позволяющее различным частям приложения, независимо от того на каком языке они написаны, взаимодействовать друг с другом (рис. 1). Даже client-side JavaScript код, выполняющийся в браузере, может общаться на той же шине (event bus). В соответствующем API поддерживается сообщения типа издатель/подписчик, точка-точка, запрос-ответ.

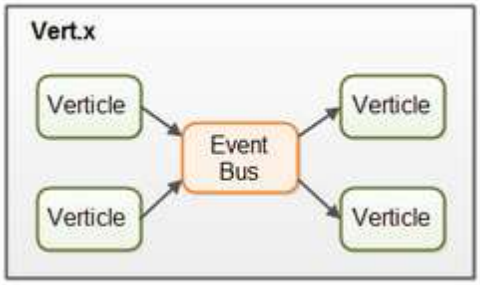

Рис. 1. Схема взаимодействия частей Vert.x приложения через Event bus

Vert.x естественно интегрируется с RxJava – популярной библиотекой для асинхронной обработки потоков данных, основанной на паттерне проектирования *Observer*.

Таким образом, в докладе рассмотрена общая информация о разработке микросервисного приложения с помощью платформы Vert.x, которая является асинхронной и неблокирующей. Следует отметить отличие ее многопоточной модели от модели, принятой во многих фреймворках, когда каждому сетевому клиенту назначается отдельных поток, что может нанести ущерб масштабируемости приложения, а также при большой нагрузке операционная система значительное время тратит на управление диспетчеризацией потоков.

Библиографические ссылки

1. Vert.x documentation [Электронный ресурс]. – Mode of access: http://vertx.io/docs/ . – Date of access: 14.03.2018.

2. A gentle guide to asynchronous programming with Eclipse Vert.x for Java developers. [Электронный ресурс]. – Mode of access: http://vertx.io/docs/guide-for-java-devs/. – Date of access: 12.02.2018.

АНАЛИТИЧЕСКАЯ СИСТЕМА ПРОГНОЗИРОВАНИЯ ОШИБОК ВЕБ-ПРИЛОЖЕНИЙ

Сеглюк И. А., Камоцкий Р. Г., Савенко А. Г.

Институт информационных технологий БГУИР, Минск, Беларусь,
e-mail: savenko@bsuir.by

Для того чтобы максимально обезопасить систему от сбоев, не достаточно иметь квалифицированный персонал, поскольку специалист попросту может не обратить внимания на факторы, указывающие на скорое возникновение ошибки.

Предлагается вариант системы, осуществляющей сбор и анализ всех необходимых данных, для последующего указания мест, где может возникнуть ошибка. Работу системы