Шаблоны, лежащие в основе java, kubernetes и современных распределенных систем

Константин Сергеевич Глумов
Инженер-программист
Альфа-Банк
Москва, Россия
glumovk@gmail.com
ORCID 0000-0000-0000-0000

Поступила в редакцию 07.11.2023
Принята 28.12.2023
Опубликована 15.03.2024

УДК 34.13.30.150
EDN FTRGFN
ВАК 4.3.1. Технологии, машины и оборудование для агропромышленного комплекса (технические науки)
OECD 02.02.AC AUTOMATION & CONTROL SYSTEMS

Аннотация

Статья посвящена исследованию шаблонов, лежащих в основе java, kubernetes и современных распределенных систем. Автором обосновывается актуальность и значимость темы исследования. Постулируется о том, что в настоящее время, шаблоны, лежащие в основе java, kubernetes и современных распределенных систем представляет собой широко распространенную платформу управления контейнерами, которая упрощает развертывание, масштабирование и управление контейнерными приложениями. Анализ научной литературы позволил заключить о том, что шаблоны – это один из самых популярных архитектурных стилей для создания облачных приложений. Они решают проблему сложности программного обеспечения за счет модульности бизнес-возможностей и замены сложности разработки эксплуатации. Вот почему ключевым условием успешного использования микросервисов является создание приложений, которые могут масштабироваться с помощью Kubernetes. Заключается о необходимости дальнейшего изучения вопроса.

Ключевые слова

шаблоны, программирование, java, kubernetes, распределенные системы, J2EE, Spring Framework, Scala, Kotlin, облачные приложения.

Введение

Как известно, традиционное объектно-ориентированное или императивное программирование сосредоточено на использовании инструкций для изменения состояния программы. С другой стороны, декларативное или функциональное программирование фокусируется на том, чего должна выполнять программа, и не беспокоится о том, как достигается этот результат, вот почему называется программированием с отслеживанием состояния.

В этой статье опишем функциональное программирование как утверждение о взаимосвязи между первой частью задачи и остальной частью задачи.

Материалы и методы исследования

Как известно, более 20 лет назад Java пошла в широкие программистские массы, чтобы в итоге стать одним из тех стержней, вокруг которых строятся стеки приложений. Однако на сегодняшний день многие люди и организации переходят на платформу Kubernetes с набором встроенных контроллеров, которые по умолчанию устанавливаются в каждое окружение платформы. Этот по-настоящему

ориентированный на STEM мозг распознает ряд шаблонов, которые управляют современными вычислениями.

Однако анализ современного опыта позволяет сделать вывод о том, что Java все еще растет. Именно инновации фреймворков и открытого исходного кода, выводящие Java за рамки J2EE, придали Java новую энергию, что говорит о важности базовой технологии и ее расширяемости.

Spring Framework изменил Java к лучшему, и это показывает непреходящую ценность глубокого ядра с расширениями, которые служат способом распространения инноваций по всему миру. Сегодня, как и вчера, важно ядро Java, но именно Spring Framework с открытым исходным кодом предлагает новые возможности. Более того, Java стала больше, чем просто языком, а Kubernetes – больше, чем просто контейнерным оркестратором. Эти инструменты с открытым исходным кодом стали платформой платформ, которые управляют шаблонами, поддерживающими новый опыт разработчиков.

Именно этот шаблон базовых технологий, фреймворков и открытого исходного кода служит основой для Kubernetes и потребности в высокораспределенных системах. В этой статье рассмотрим его детальнее.

Результаты и обсуждение

Spring Framework предоставляет собой комплексную модель программирования и настройки для современных корпоративных приложений на базе Java – на любой платформе развертывания. Ключевым элементом Spring является инфраструктурная поддержка на уровне приложений: Spring фокусируется на «настройке» корпоративных приложений, чтобы команды могли сосредоточиться на бизнес-логике на уровне приложений без ненужных привязок к конкретным средам развертывания. Кстати, то же относится к Scala, но Kotlin.

Scala и новые языки, такие как Kotlin, приобрели значимость, потому что это, прежде всего, функциональные языки, более выразительные, чем предписывающие. Следует согласиться с мнением К. Дэвис о том, что популярность Kotlin заключается в том, что это «в первую очередь функциональный язык программирования, который основан на математических функциях, предназначенных для обработки символьных вычислений и приложений обработки списков» (Davis, 2019).

В настоящее время языки предназначены для нового мира распределенных систем, которые должны полагаться на глубокие технические возможности распределенных архитектур.

Так, известно, что с помощью Kubernetes разработчики могут создавать облачные приложения, используя библиотеку интерфейсов прикладного программирования (API), а также инструменты для создания приложений. К. Дэвис отмечает, что переход к облачным приложениям приводит к появлению шаблонов, с которыми должен быть знаком каждый. Среди них шаблоны:

- автоматического отключения – обертывание монитора, отслеживающего сбой;
- обнаружения сервисов – автоматическое обнаружение сервисов и устройств;
- повторных попыток – повторные попытки после кратковременной потери сети.

Поясним. Шаблоны – это способ повторного использования архитектур. Вместо того чтобы создавать сквозную архитектуру самостоятельно, можно использовать существующие шаблоны Kubernetes и, таким образом, убедиться, что все элементы будут работать должным образом. Отметим, что первоначально разработанная Kubernetes была платформой оркестровки контейнеров с открытым исходным кодом, предназначенной для автоматизации развертывания, масштабируемости и управления контейнерными приложениями. Фактически, Kubernetes де-факто стал стандартом для оркестровки контейнеров и является флагманским проектом Cloud Native Computing Foundation (CNCF), поддерживаемым такими крупными игроками, как Google, AWS, Microsoft, IBM, Intel, Cisco, и Red Hat. На рисунке 1 показано сочетание навыков, необходимых для создания хороших облачных приложений, и показано, как подходят шаблоны Kubernetes.

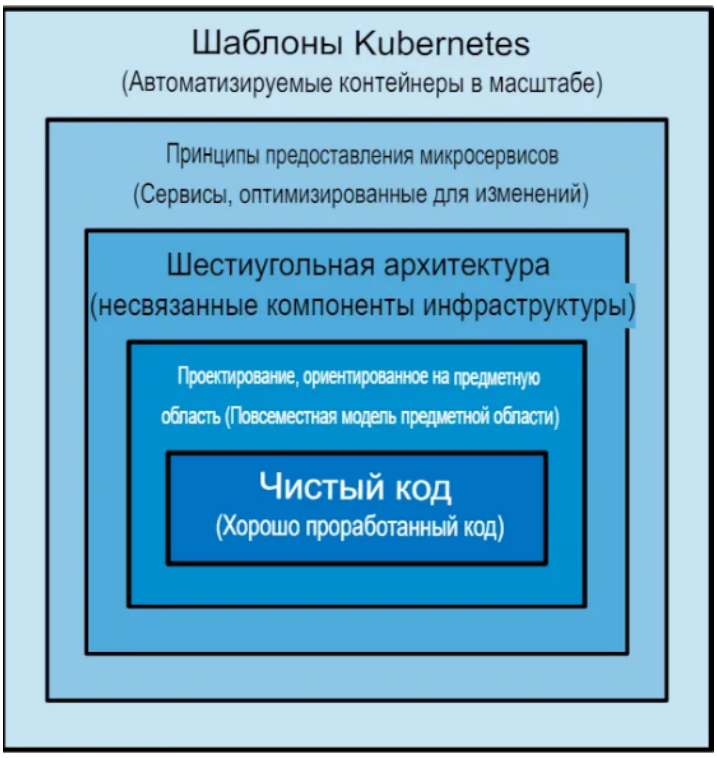

Рисунок 1. Путь к облачной среде. Использовано с разрешения Kubernetes

Kubernetes работает на архитектуре клиент-сервер, включающей плоскость управления и набор узлов, на которых запускаются контейнеры.

Заметим, что использование многоконтейнерных модулей в Kubernetes позволяет улучшить совместную работу и совместное использование ресурсов между контейнерами в рамках одного модуля. Рассмотрим некоторые фагменты кода, демонстрирующие различные шаблоны работы с несколькими контейнерами: шаблон Sidecar (рис. 2); шаблон Ambassador (рис. 3); шаблон адаптера (рис. 4).

```
01  apiVersion: v1
02  kind: Pod
03  metadata:
04    name: multi-container-pod
05  spec:
06    containers:
07    - name: main-container
08      image: main-image:latest
09      ports:
10      - containerPort: 80
11    - name: sidecar-container
12      image: sidecar-image:latest
```

Рисунок 2. Шаблон Sidecar

Контейнер sidecar работает параллельно с контейнером основного приложения, предоставляя дополнительные функциональные возможности без изменения основного приложения.

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: ambassador-pod
spec:
  containers:
  - name: main-container
    image: main-image:latest
    ports:
    - containerPort: 80
  - name: ambassador-container
    image: ambassador-image:latest
    ports:
    - containerPort: 8080
```

Рисунок 3. Шаблон Ambassador

В этом шаблоне контейнер ambassador действует как посредник, обрабатывая коммуникационные и сетевые задачи от имени основного приложения. Контейнер-адаптер преобразует или адаптирует данные до того, как они попадут в основное приложение, обеспечивая плавную точку интеграции.

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: adapter-pod
spec:
  containers:
  - name: main-container
    image: main-image:latest
    ports:
    - containerPort: 80
  - name: adapter-container
    image: adapter-image:latest
    ports:
    - containerPort: 9090
```

Рисунок 4. Шаблон адаптера

Так, Kubernetes упрощает развертывание и эксплуатацию приложений в архитектуре микросервисов. Таким образом, группы разработчиков могут развертывать свои приложения и поручать Kubernetes управление следующими элементами:

– контроль потребления ресурсов приложением или командой;
– равномерное распределение нагрузки приложений на инфраструктуру хоста;
– автоматическая балансировка нагрузки запросов на разных экземплярах приложения;
– мониторинг потребления ресурсов и ограничений для автоматического выключения и перезапуска приложений, которые потребляют слишком много ресурсов;
– перемещение экземпляра приложения с одного хоста на другой в случае нехватки ресурсов на хосте или в случае его смерти;
– автоматическая мобилизация дополнительных ресурсов, которые становятся доступными при добавлении нового хоста в кластер.

В этой связи отметим, что шаблоны – это инструменты, необходимые разработчикам Kubernetes. Они рассказывают им, как создать систему:

1. Базовые шаблоны охватывают фундаментальные концепции Kubernetes. Они представляют основные принципы и методы, которые необходимо применять для создания облачных приложений на основе контейнеров.
2. Поведенческие модели располагаются поверх базовых моделей и дополнительно детализируют концепции, позволяя управлять многими типами взаимодействий между контейнерами и платформами.
3. Структурные шаблоны позволяют упорядочивать контейнеры в модуле Kubernetes.
4. Шаблоны конфигурации используются для управления множеством способов настройки приложения в Kubernetes. Они включают в себя шаги, необходимые для подключения приложений к их настройке.
5. Расширенные шаблоны предназначены для расширенных концепций, таких как способы расширения платформы или создания образов контейнеров непосредственно в кластере.

Шаблоны Kubernetes представляют многоразовые шаблоны и принципы проектирования и реализации облачных приложений на Kubernetes, к примеру Red Hat® OpenShift® – это платформа Kubernetes, разработанная для предприятий. Он предлагает разработчикам среды самообслуживания для создания приложений, а также функции для автоматизации работы всего стека на любом типе инфраструктуры (Hass, 2020).

Red Hat OpenShift включает в себя множество дополнительных технологий, которые делают Kubernetes мощным и жизнеспособным инструментом для бизнеса, в том числе: ведение реестра, сетевое взаимодействие, телеметрию, безопасность, автоматизацию и услуги.

С помощью Red Hat OpenShift разработчики могут создавать контейнерные приложения, размещать их и развертывать в облаке, обеспечивая при этом эффективную масштабируемость, контроль и организацию, чтобы идеи могли быстро стать реальностью. Облачные технологии базируются на виртуальном ПО, объединяющем ресурсы нескольких серверов. Вследствие этого, по сути – это новая технологическая парадигма. Если, например, в традиционной корпоративной IT-среде нужно будет сделать сервер нового приложения, на это может уйти не одна неделя времени. Ведь нужно купить новый сервер, затем перенастроить все ПО именно под него. Когда же речь идет об облачной среде, то в ней объем сервера виртуален. В результате и настроить, и запустить его в работу можно буквально за несколько минут. А приложения с такого сервера запускаются вообще практически на автомате. При этом в данном случае клиент оплачивает лишь то виртуальное пространство, которым действительно пользуется (в случае с традиционным сервером нужно платить за весь его объем). Возможности облачных технологий действительно безграничны, а сервисы – многообразны. Здесь можно просто хранить какие-то данные, а можно при необходимости использовать в собственных целях сложнейшие и притом безопасные IT-структуры.

Заключение

Вышесказанное позволяет сделать объективное заключение о том, что kubernetes несомненно развивался в течение всего времени своего существования. Однако такой быстрый рост также иногда приводит к трудностям. Хотя разработчики любят технологии с открытым исходным кодом, такие как Kubernetes, за потенциальную скорость предлагаемых ими инноваций, иногда бывает также, что слишком много инноваций создает некоторую путаницу (особенно когда код Kubernetes Central работает быстрее, чем пользователи успевают за его разработкой). Однако, все же, разработчики изо дня в день, используют все больше примитивных типов данных в Java и со временем эти новые примитивы порождают новые способы решения проблем, и некоторые из этих повторяющихся решений становятся шаблонами. Таким образом, шаблоны значительно упрощают жизнь Java-разработчику при работе с Kubernetes.

Список литературы

1. Academy, F. How is Java different from other Programming Languages? (FITA Academy Student Video Review,2021), https://www.fita.in/how-java-is-different-from-other- programming-languages/
2. Davis C. Cloud Native Patterns. Released May. 2019. 400 p.
3. Everything computer science CS Java. (2016), https://every-thingcomputerscience.com/programming/Java.html
4. Hachadi, Z. Java Web Development Springboot Security..
5. Hass R., Ibram B. Templates for developing your own cloud applications. 2020. 320 p
6. Interview Bit Major Features of Java Programming Language. (2023), https://www.interviewbit.com/blog/features-of-java/

The patterns underlying java, kubernetes, and modern distributed systems

Konstantin S. Glumov
Software Engineer
Alfa-Bank
Moscow, Russia
glumovk@gmail.com
ORCHID 0000-0000-0000-0000

Received 07.11.2023
Accepted 28.12.2023
Published 15.03.2024

UDC 34.13.30.150
EDN FTRGFN
VAK 4.3.1. Technologies, machinery and equipment for the agro-industrial complex (technical sciences)
OECD 02.02.AC AUTOMATION & CONTROL SYSTEMS

Abstract

The article is devoted to the study of the patterns underlying java, kubernetes and modern distributed systems. The author substantiates the relevance and significance of the research topic. It is postulated that currently, the patterns underlying java, kubernetes and modern distributed systems represent a widespread container management platform that simplifies the deployment, scaling and management of container applications. An analysis of the scientific literature has led to the conclusion that templates are one of the most popular architectural styles for creating cloud applications. They solve the problem of software complexity by modularizing business capabilities and replacing development complexity with operational complexity. That's why the key to successful use of microservices is to create applications that can scale with Kubernetes. It is about the need for further study of the issue.

Keywords

templates, programming, java, kubernetes, distributed systems, J2EE, Spring Framework, Scala, Kotlin, cloud applications.

References

1. Academy, F. How is Java different from other Programming Languages? (FITA Academy Student Video Review,2021), https://www.fita.in/how-java-is-different-from-other- programming-languages/
2. Davis C. Cloud Native Patterns. Released May. 2019. 400 p.

3. Everything computer science CS Java. (2016), https://every-thingcomputerscience.com/programming/Java.html
4. Hachadi, Z. Java Web Development Springboot Security..
5. Hass R., Ibram B. Templates for developing your own cloud applications. 2020. 320 p
6. Interview Bit Major Features of Java Programming Language. (2023), https://www.interviewbit.com/blog/features-of-java/