СОЗДАНИЕ РАСПРЕДЕЛЕННЫХ СИСТЕМ

и. т. н. ***Бакунова О. М.,***
м. т. н. ***Калитеня И. Л.,***
студент ***Петрович А. С.,***
студент ***Думиков А. А.,***
студент ***Буркин А. В.,***
студент ***Протько Д. В.***

Республика Беларусь, г. Минск, Институт информационных технологий Белорусского государственного университета информатики и радиоэлектроники

ARTICLE INFO
Received 03 March 2018
Accepted 17 March 2018
Published 12 April 2018

KEYWORDS
Services,
microservices,
system design,
software architecture,
distributed systems

ABSTRACT
One of the key steps of creation a software is planning its architecture. Neglect of this stage leads to a significant increase of development costs and the worst software performance. Usually, each software requires its own, unique approach to development, but effective planning is impossible without knowing modern ways to design software. One of the most effective approach is to create or rebuild software as a distributed system.

Введение. Распределенная система — сложное программное обеспечение, которое выполняется на нескольких компьютерах, имеющих различное техническое оснащение, операционные системы, предустановленные программы. В данной статье описываются принципы и методы построения распределенных систем.

Концепция распределенной системы была разработана более двадцати лет назад, но стала набирать популярность и приобрела широкое распространение только в последние годы.

Исторически сложилось, что потребность в программном обеспечении растет быстрее, чем вычислительная мощность компьютеров. На определенном этапе это привело к тому, что затраты на создание и обслуживание одного мощного компьютера превысили доход от размещения на нем программного обеспечения. Тогда была разработана концепция распределенной системы, в которой различные части программного обеспечения выполняются на различных компьютерах. Прообразом распределенной системы может служить клиент-серверная архитектура программного обеспечения, при которой часть функций выполняет компьютер пользователя — клиент, — а другую — сервер поставщика услуг.

Результаты и обсуждение. Распределенные системы определены как набор небольших независимых модулей, которые функционируют одновременно на различных вычислительных машинах и служат решению общей задачи. Таким образом возможно легко масштабировать любой модуль распределенной системы: сложные модули запускают на мощных компьютерах, тогда как сравнительно простые — на более слабых.

Вопреки положительным качествам, разработка распределенной системы имеет большую техническую сложность, так что может быть оправдана только при необходимости создать такое программное обеспечение, которое выдержит большие нагрузки. Примером качественной распределенной системы может служить платформа потокового вещания в сети интернет «Netflix». Чтобы обеспечить надежность сервиса, при разработке программного средства компания «Netflix» придерживалась микросервисной архитектуры. Применение данной архитектуры — современный способ построения распределенной системы.

Бесспорная эффективность данного подхода подтверждается опытом многих известных корпораций, таких как Google, Microsoft, Amazon и других.

Понятием, противоположным распределенной системе, является так называемый монолит. Монолитное программное обеспечение целиком хранится в единственном вычислительном центре. Согласно исследованию Сэма Ньюмэна, представленному в его работе

«Создание микросервисов», доля монолитного программного обеспечения снижается с каждым годом, и особенно стремительно – в последние пять лет. Предприятия, которым недостает вычислительной мощности, активно приводят свое монолитное программное обеспечение к форме распределенной системы.

Главные принципы создания качественной распределенной системы таковы: моделирование вокруг бизнес-процессов, сокрытие деталей внутренней реализации, децентрализация, изоляция сбоев, независимое выполнение, обширные возможности для наблюдения, повсеместная автоматизация. Строгое следование этим принципам позволяет создать систему, которая будет работать стабильно даже при выведении из строя сразу нескольких модулей.

Согласно закону, выведенному Мелвином Конвеем в 1968 году в его публикации «How Do Committees Invent», структура системы, которую разрабатывает организация, должна повторять структуру самой организации. Именно этот закон описывает первый принцип разработки распределённой системы – моделирование вокруг бизнес-процессов. В реальных компаниях различные подразделения мало связаны между собой, но сильно связаны внутри себя, так что полезно создать распределенную систему, повторяющую их структуру.

Детали реализации каждого модуля распределённой системы должны быть сокрыты от других модулей. Разделённые модули должны взаимодействовать только посредством заданного интерфейса. В противном случае это может привести к «склеиванию», то есть превращению распределённой системы обратно в монолит. Однако инженеру, который разрабатывает архитектуру распределённой системы, придётся решать ещё одну важную задачу, относящуюся к принципу сокрытия реализации. Известно, что для выполнения одной и той же задачи некоторые языки и платформы программирования могут подходить лучше прочих. В то же время одни операционные системы будут справляться с выполнением одного и того же программного кода лучше других. Так что может показаться, что наиболее эффективной будет та распределённая система, в которой каждый из модулей был разработан и выполняется на наиболее подходящей для этого программно-аппаратной платформе. Однако подобное разнообразие уменьшает надёжность программного продукта целиком. Так, например, если какой-то из модулей написан на языке программирования, эффективном для данной конкретной задачи, но малораспространённом, архитектор может столкнуться с так называемым «фактором автобуса» или «фактором кирпича». В области разработки программного обеспечения «фактор кирпича» — это мера сосредоточения информации среди отдельных членов проекта. Этот фактор показывает количество разработчиков программного продукта, после потери которых проект не может быть продолжен. Например, проект может содержать такую информацию, с которой оставшиеся разработчики не смогут разобраться. Высокий «фактор кирпича» проекта означает, что проект будет устойчиво развиваться, если его покинет даже большое количество программистов. Таким образом, архитектору программного продукта придётся искать баланс между «фактором кирпича» и эффективностью исполнения кода.

Децентрализация и изоляция сбоев**.** Отсутствие «оркестратора» – центра, управляющего всеми модулями системы, – означает, что сбой любого модуля не должен привести к краху целой системы. Это также означает, что при построении распределённых систем мы стремимся избежать расположения их на единой платформе или даже одной географической точке в целях предотвращения потери работоспособности системы целиком ввиду отказа каких-либо физических серверов. Также система должна быть построена таким образом, чтобы нестабильная работа одного из модулей не могла повлиять на систему целиком.

Когда система распределена и децентрализована, каждый её модуль можно запустить вне зависимости от работы других модулей. Это привносит одно из ключевых преимуществ для команд разработчиков – модуль системы, который разрабатывается той или иной командой, может выполняться и изменяться вне зависимости от других модулей. Но если одна из команд вносит ключевое изменение в свой участок приложения, она должна позаботиться о том, чтобы предыдущий вариант был доступен достаточно долгое время, чтобы прочие команды подстроили свои модули под это изменение. Кроме того, автоматизация выполнения базовых проверок каждого из модулей может существенно сократить затраты на разработку программного комплекса.

Концепция распределённых систем подразумевает, что разработчики и архитекторы неизбежно столкнутся с множеством ошибок. Но в отличие от монолитных систем, ошибки в распределённых системах не являются критическими. Как описано выше, любой сбой любого модуля не должен влиять на работу приложения целиком. Однако для команд, участвующих в

разработке и поддержке MSA, важно быстро распознавать и реагировать на появление таких ошибок. Для достижения этой цели важно правильно настроить системы мониторинга.

Выводы. При чётком следовании концепции построения распределённых систем разрабатываемые сложные технические продукты могут эффективно выполняться на различных платформах одновременно, что значительно уменьшает стоимость их эксплуатации и в то же время упрощает дальнейшую поддержку. Однако эффективность такого подхода к разработке следует оценивать в каждом конкретном случае отдельно.

ЛИТЕРАТУРА

1. С. Ньюмен Создание микросервисов. — СПб.: Питер, 2016. — 304 с..
2. Электронный ресурс. Режим доступа:http://www.melconway.com/Home/pdf/committees.pdf