ГЛАВА 1

Текущее положение дел в наблюдаемости

> История — не прошлое, а карта прошлого, нарисованная с конкретной точки зрения, которая должна быть полезной для современного путешественника.
>
> *Генри Гласси,*
> *американский историк*[1]

Эта книга посвящена сложным проблемам, присущим крупномасштабным распределенным компьютерным системам, и применению OpenTelemetry для решения этих проблем. Современная разработка ПО придает огромное значение опыту конечных пользователей, а конечные пользователи требуют моментального быстродействия. Опросы показывают, что пользователи покидают сайты интернет-магазинов, загрузка которых занимает более 2 секунд (https://oreil.ly/tZ9tY). Вероятно, вы потратили немало времени на попытки оптимизации и отладки проблем быстродействия приложений, и если у нас с вами есть хоть что-то общее, вас наверняка раздражало, насколько неэлегантным

[1] Henry Glassie, «Passing the Time in Ballymenone: Culture and History of an Ulster Community» (Philadelphia: University of Pennsylvania Press, 1982).

и неэффективным может быть этот процесс. Либо не хватает данных, либо их слишком много, а в имеющихся данных полно противоречий или неоднозначных измерений.

Инженеры также работают в условиях жестких требований ко времени безотказной работы. Это означает, что любые проблемы должны выявляться и исправляться до того, как произойдет сбой, — вместо того, чтобы дожидаться отказа системы. А это подразумевает быстрый переход от диагностики к исправлению. Для этого вам понадобятся данные.

Но вам нужны не какие-то случайные данные, вам нужны *коррелированные данные*, то есть данные, уже упорядоченные и готовые к анализу компьютерной системой. Как вы вскоре увидите, данные с таким уровнем организации не бывают доступны с самого начала. А по мере того как системы масштабируются и становятся более разнородными, найти данные, необходимые для анализа проблемы, будет еще сложнее. Если когда-то задачу можно было сравнить с поиском иголки в стоге сена, то теперь она начинает напоминать поиски иголки в стоге иголок.

OpenTelemetry решает эту проблему. Преобразуя отдельные журналы, метрики и данные трассировки в связный унифицированный граф с информацией, OpenTelemetry закладывает фундамент для следующего поколения средств поддержки наблюдаемости. А поскольку технология OpenTelemetry уже получила широкое распространение в отрасли разработки ПО, следующее поколение средств создается уже во время написания этой книги.

Такие разные времена

Развитие технологий происходит волнообразно. Когда мы пишем этот раздел в 2024 году, в области наблюдаемости как раз прокатывается первое настоящее цунами за последние как минимум 30 лет. Вы выбрали удачное время, чтобы прочитать эту книгу и взглянуть на ситуацию под новым углом.

Приход облачных вычислений и облачно-ориентированных систем привел к тектоническим сдвигам в практике построения и управления сложными программными системами. Однако кое-что осталось неизменным: программы выполняются на компьютерах, и вы должны понимать, что делают эти компьютеры, чтобы понимать вашу программную систему. Как бы облачные технологии ни стремились к формированию абстракций как фундаментальных единиц вычислений, ваши последовательности нулей и единиц все еще соответствуют битам и байтам.

Где бы ни выполнялась ваша программа — на межрегиональном кластере Kubernetes или на ноутбуке, вы будете задавать себе одни и те же вопросы:

- «Почему она так медленно работает?»
- «Почему она расходует столько памяти?»
- «Когда это началось?»
- «В чем корень проблемы?»
- «Как это исправить?»

Астроном и популяризатор науки Карл Саган (Carl Sagan) однажды сказал: «Чтобы понять настоящее, необходимо знать прошлое»[1].

Бесспорно, это высказывание применимо и здесь: чтобы увидеть, почему новый подход к наблюдаемости так важен, сначала необходимо познакомиться с традиционным подходом к наблюдаемости и его ограничениями.

Похоже на повторение известных всем истин. Однако неразбериха с наблюдаемостью существует уже так давно, что многие из нас

[1] Карл Саган (автор и ведущий), в сериале «Cosmos: A Personal Voyage», сезон 1, эпизод 2, «One Voice in the Cosmic Fugue», продюсер Эдриэн Мэлоун (Adrian Malone) (Arlington, VA: Public Broadcasting Service, 1980).).

успели обзавестись целой кучей предубеждений и стереотипов. Итак, даже если вы эксперт — и *особенно* если вы эксперт, — важно взглянуть на ситуацию свежим взглядом.

Наше путешествие начнется с определения основных понятий, которые будут использоваться в книге.

Наблюдаемость: основные понятия

Прежде всего, что наблюдает наблюдаемость? В контексте этой книги мы наблюдаем за распределенными системами. *Распределенной системой* называется система, компоненты которой находятся на разных компьютерах в сети. Эти компьютеры взаимодействуют и координируют свои действия, передавая сообщения друг другу[1].

Существует много разновидностей компьютерных систем. Ниже перечислены те, которые рассматриваются в книге.

Что значит «распределенная»?

Распределенные системы — не просто приложения, уже работающие в облаке, микросервисы или приложения Kubernetes. Макросервисы (или «монолиты»), использующие сервисно-ориентированную архитектуру, клиентские приложения, взаимодействующие с бэкендом, мобильные и веб-приложения — все они в какой-то мере являются распределенными и извлекают пользу из анализа производительности.

На самом высоком уровне распределенная система состоит из ресурсов и транзакций.

---

[1] Andrew S. Tanenbaum and Maarten van Steen, «Distributed Systems: Principles and Paradigms» (Э. Таненбаум, М. ван Стеен. «Распределенные системы. Принципы и парадигмы». СПб., издательство «Питер»).

Ресурсы

Физические и логические компоненты, образующие систему. *Физические компоненты* (серверы, контейнеры, процессы, ОЗУ, процессор, сетевые адаптеры) относятся к ресурсам. *Логические компоненты* (клиенты, приложения, конечные точки API, базы данных и распределители нагрузки) также являются ресурсами. Короче говоря, к ресурсам относится все, из чего непосредственно строится система.

Транзакции

Существуют запросы, которые координируют и используют ресурсы, необходимые системе для выполнения работы по поручению пользователя. Обычно транзакция запускается человеком, атрибут ожидает ее завершения. Бронирование места на авиарейсе, организация совместного использования автомобиля, загрузка веб-страниц — все это примеры транзакций.

Как анализировать эти распределенные системы? Никак, если они не выдают данные, которые принято называть «*телеметрия*». Телеметрия — это данные, описывающие текущее функционирование системы. Без телеметрии система представляет собой большой таинственный черный ящик.

Многим разработчикам слово «*телеметрия*» кажется непонятным. Этот термин слишком многозначный. Важно, как в рамках этой книги, так и в мониторинге систем в целом, разделять пользовательскую телеметрию и телеметрию производительности:

Пользовательская телеметрия (user telemetry)

Относится к данным, описывающим взаимодействие пользователя с системой через клиента: нажатия кнопок, продолжительность сеанса, информация о хосте клиента и т. д. По этим данным можно понять, как пользователи взаимодействуют с сайтом интернет-магазина, или получить информацию

о статистике распределения версий браузеров при обращении к веб-приложению.

Телеметрия производительности (performance telemetry)

Эта телеметрия обычно не используется для анализа поведения пользователя; вместо этого она предоставляет оператору статистическую информацию о поведении и быстродействии системных компонентов. Данные производительности в распределенной системе могут поступать из многих источников; они предоставляют разработчикам «след из хлебных крошек», по которому можно связать причину со следствием.

Проще говоря, пользовательская телеметрия сообщает, как долго пользователь удерживает указатель мыши над кнопкой оформления заказа в приложении интернет-магазина. Телеметрия производительности сообщает, сколько времени заняла загрузка этой кнопки и какие при этом были использованы программы и ресурсы.

В основе пользовательской телеметрии и телеметрии производительности лежат разные типы сигналов. *Сигнал* представляет собой особую форму телеметрии. Журналы событий — один из видов сигналов. Системные метрики — другой. Непрерывное профилирование — еще один. Каждый из этих типов сигналов служит определенной цели, и они не заменяют друг друга. Вы не сможете сделать вывод обо всех событиях пользовательского взаимодействия, просто просматривая системные метрики, и не сможете узнать о степени загруженности системы, просматривая журналы транзакций. Для достижения глубокого понимания системы в целом вам понадобятся разные виды сигналов.

Каждый сигнал состоит из двух частей: *инструментального кода* (кода, выдающего телеметрические данные) в самих программах и *системы передачи* для отправки данных по сети аналитической системе, где и выполняется фактическое наблюдение.

Здесь проявляется важное различие: телеметрию часто путают с анализом, но важно понимать, что система, выдающая данные, и система, которая анализирует данные, это не одно и то же. *Телеметрией* называются сами данные, а *анализ* — это то, что вы делаете с этими данными.

Наконец, телеметрия в совокупности с анализом сообщает системе свойство *наблюдаемости*. Понимание того, как лучше объединить эти две составляющие в полезную систему, — основная тема этой книги.

НАБЛЮДАЕМОСТЬ КАК ПРАКТИКА

Свойство наблюдаемости — это не просто совокупность телеметрии и анализа; это организационная практика, сходная с практикой DevOps. Во многих отношениях наблюдаемость является фундаментом современных практик разработки — она лежит в основе всего, что мы делаем, от непрерывной интеграции и развертывания (CI/CD) до хаос-инженерии и производительности разработки и многого другого. Ее источники настолько же обширны и разнообразны, как и команды разработчиков и программные продукты, и эти данные можно собирать, анализировать и использовать для непрерывного улучшения работы. Надеемся, эта книга вооружит вас фундаментальными знаниями, необходимыми для внедрения практики наблюдаемости в вашей организации на базе OpenTelemetry!

Краткая история телеметрии

Интересный факт: термин «телеметрия» появился из-за того, что первые системы удаленной диагностики передавали данные по телеграфным линиям. Когда мы слышим слово «телеметрия», нам приходят на ум космические программы 1950-х годов, но если бы практика телеметрии вела свою историю с той эпохи, она бы

называлась «радиометрией». Телеметрия разрабатывалась для слежения за электростанциями и сетями электроснабжения — ранним, но важным прототипом распределенных систем!

Конечно, компьютерная телеметрия появилась позднее. История пользовательской телеметрии и телеметрии производительности отражает изменения в работе с программами, непрерывный рост вычислительных мощностей и пропускной способности сетей, всего, что стоит у истоков современных процессов. Знание того, как появлялись сигналы компьютерной телеметрии, и направление их эволюции становятся важной частью понимания пределов их возможностей в настоящее время.

Первой и самой долгоживущей формой телеметрии было *ведение журнала* (*логирование*). В *журналах* сохраняются текстовые сообщения, описывающие состояние системы или сервиса и предназначенные для восприятия людьми. Со временем разработчики и операторы совершенствовали механизмы сохранения и поиска журнальных сообщений, создавая специализированные базы данных, хорошо подходящие для полнотекстового поиска.

Хотя в журналах хранится информация об отдельных событиях и моментах работы системы, для понимания того, как система изменяется со временем, требуется больше данных. В журнале можно увидеть, что попытка записи в файл окончилась неудачно, потому что на носителе не хватило свободного места, но разве не лучше, если бы вы могли отслеживать объем свободного пространства и освободить его *до того*, как произойдет ошибка?

Метрики представляют собой статистические представления состояния системы и потребления ресурсов. Они идеально подходили для своей роли. Добавление метрик позволяло строить механизмы оповещения о данных, не ограничивающиеся сообщениями об ошибках и генерацией исключений.

С развитием современного интернета системы становились более сложными, а быстродействие — более значимым.

Появилась третья форма телеметрии: *распределенная трассировка* (distributed tracing). По мере того как транзакции стали включать все больше операций и больше машин, локализация источника проблемы начинает приобретать все более важную роль. Вместо того чтобы рассматривать отдельные события (записи в журнале), трассировочная система рассматривала целые операции и их объединения для формирования транзакций. У операций есть определенное начальное и конечное время. Они также связываются с определенным местом: на какой машине была выполнена конкретная операция? Трассировка этой информации позволяет проследить источник задержки до конкретной операции или конкретного компьютера. Однако из-за ресурсных ограничений системы трассировки обычно работали с малочисленной выборкой и в итоге сохраняли только небольшую часть от общего числа транзакций и выдавали лишь самый базовый анализ быстродействия.

Три вкладки в браузере

Хотя существуют и более полезные формы телеметрии, первичность этих трех систем — журналов, метрик и трассировки — привела к концепции, которая в наши дни известна под названием трех столпов наблюдаемости[1].

Концепция «трех столпов» отлично описывает современную практику наблюдаемости, но в действительности это далеко не лучший способ *проектирования* телеметрической системы!

Традиционно каждая форма наблюдаемости — телеметрия плюс анализ — строилась как полностью отдельная, изолированная система, что показано на рис. 1.1.

[1] Cindy Sridharan, «Distributed Systems Observability» (Sebastopol, CA: O'Reilly, 2018).

Рис. 1.1. Столп наблюдаемости

Система ведения журнала состоит из кода ведения журнала, системы передачи журналов и средств анализа журнала. Система метрик состоит из кода сбора метрик, системы передачи метрик и средств анализа метрик. То же относится к трассировке — отсюда три столпа, представленные на рис. 1.2.

Рис. 1.2. Три столпа наблюдаемости

Перед вами простейшая вертикальная интеграция: каждая система строится для определенной цели, от начала и до конца. Неудивительно, что наблюдаемость строилась именно по такой схеме — она развивалась со временем, и новые части добавлялись по мере надобности. Иначе говоря, такая структура наблюдаемости не объяснялась ничем, кроме исторической случайности. Простейший способ реализации системы ведения журнала или

метрической системы — изолированная реализация в виде автономной системы.

Таким образом, хотя термин «три столпа» объясняет архитектуру традиционной наблюдаемости, он также создает проблемы — может показаться, что такая архитектура хороша! На самом деле нет. Кому-то это покажется самонадеянным, но я предпочитаю другое выражение: «три вкладки в браузере». Потому что именно это вы получаете в итоге.

Возникающие трудности

Проблема в том, что наши системы не формируются из задач ведения журнала или нахождения метрик. Они формируются из транзакций и ресурсов. Когда возникает проблема, есть только два аспекта, которые можно менять: разработчики могут изменить то, что делают транзакции, а операторы могут изменить состав доступных ресурсов. И это все.

Но как известно, дьявол кроется в деталях. Простая, изолированная ошибка может быть ограничена рамками одной транзакции. Но многие проблемы в ходе эксплуатации возникают из-за специфики взаимодействий многих конкурентных транзакций.

В наблюдении за реальными системами значительное место занимает выявление закономерностей нежелательного поведения с последующей экстраполяцией; цель — определить, как некоторые комбинации транзакций и потребления ресурсов порождают эти закономерности. А это весьма непросто! Очень трудно предсказать, как транзакции и ресурсы будут взаимодействовать в реальном мире. Тесты и маломасштабные развертывания не всегда хорошо подходят для этой задачи, потому что проблемы, которые вы пытаетесь устранить, проявляются только в рабочих средах. Они проявляются как побочные эффекты, и они обусловлены взаимодействиями реальных пользователей системы с вашей рабочей средой.