ГЛАВА 1

Сначала главное

Добро пожаловать! Давайте разберемся, что такое обеспечение надежности информационных систем (site reliability engineering, SRE) и как оно появилось.

Что такое SRE

Вам может встретиться несколько определений понятия «обеспечение надежности информационных систем». Вот лучшее из тех, что мне удалось сформулировать за многие годы:

> Обеспечение надежности информационных систем — это инженерная дисциплина, призванная помочь организации стабильно достигать соответствующего уровня надежности своих систем, сервисов и продуктов.

Озвучивая это определение аудитории, я обычно говорю, что в нем есть как минимум три слова, наличие которых, если их правильно понять, даст достаточно полное представление о SRE. Если появляется возможность, я спрашиваю у слушателей: «Какие три слова, по вашему мнению, самые важные в этом определении?» Пожалуйста, не стесняйтесь, остановитесь, перечитайте определение выше и ответьте для себя на этот вопрос, прежде чем продолжить.

Я задаю его не только потому, что мне нравится общение с аудиторией, но и потому, что он позволяет протестировать слушателей и сделать выводы. В главе 4 я подробнее остановлюсь на том, что можно узнать с помощью такой диагностики. А пока посмотрим, какие три слова я бы выбрал в первую очередь, если бы спросили меня.

Надежность

Весьма очевидное первое предположение, правда? Надежность занимает центральное место во всем, что мы делаем в SRE (да, это прямо в названии указано). Один из способов подчеркнуть важность надежности — отметить, что компания может потратить миллионы в местной валюте на создание самого лучшего программного обеспечения с самыми новомодными характеристиками, нанять отличных специалистов по продажам, которые будут его продавать, собрать отличную команду специалистов из службы техподдержки, чтобы поддерживать его работу, и т. д., но, если программное обеспечение не будет работать, когда клиент попытается им воспользоваться, все деньги и усилия окажутся выброшены на ветер (или смыты в унитаз, в зависимости от того, какая метафора вам больше нравится).

Когда возникают проблемы с надежностью, компания может терять:

Деньги

Это особенно актуально, если вышедшая из строя система критически важна для получения прибыли.

Время

Вместо выполнения запланированной работы сотрудники решают проблемы, вызвавшие сбой.

Репутацию

Люди не захотят пользоваться услугами, которые им кажутся некачественными, и с радостью уйдут к конкуренту.

Здоровье

Если условия работы постоянно экстремальные, если дежурных регулярно будят, если сотрудникам постоянно приходится тратить время на работу, а не на друзей или семью, — это может серьезно испортить их здоровье.

Людей

Специалисты в этой отрасли общаются друг с другом. Если станет известно, что у вас на работе постоянный аврал, вам будет очень трудно нанимать новых сотрудников.

Соответствие

Полагаю, одна из ключевых идей, выделяемых или подчеркиваемых SR-инженерами в обсуждениях методов эксплуатации, — что стопроцентная надежность является желательной или хотя бы возможной целью лишь в редчайших случаях. Во многих ситуациях это невозможно, ведь в нашем взаимосвязанном мире очень высока вероятность того, что зависимости не будут на 100 % надежны.

Добиться более высокого уровня надежности, чем существующий у зависимостей, порой (но не всегда) можно с помощью продуманных планирования и кода.

Вместо этого область SRE сосредотачивается на таких методах, как показатели уровня обслуживания/цели уровня обслуживания (service level indicators, SLIs/service level objectives, SLOs)[1], чтобы помочь вам определить соответствующий уровень надежности системы, передать информацию об этом и работать над его достижением.

Стабильность

Это слово вошло в определение позже остальных, когда стало ясно, что для достижения успеха методы работы эксплуатации должны быть стабильными. Стабильность напоминает о проблеме «потери здоровья», связанной с надежностью. Надежные системы создаются людьми. Если люди в вашей компании выгорели, истощены, не имеют возможности общаться со своими близкими вне работы или позаботиться о самих себе, они не смогут создавать надежные системы. Многие узна*ют* это на собственном опыте; пожалуйста, не становитесь одним из них, если можете этого избежать.

(Дополнительные слова)

В определении есть еще несколько слов, которые я пока просто перечислю, предвосхищая обсуждение в главе 4: *инженерия*, *дисциплина*, *помощь* и *организация*. До скорой встречи в четвертой главе!

История происхождения

Я считаю, что полезно знать о происхождении области SRE и о том, как она возникла, когда я работал в корпорации Google (примерно в 2003 году), однако не мне рассказывать эту историю. Бен Трейнор Слосс, создатель направления SRE, излагает официальную версию в книге *Site Reliability Engineering* (ее еще называют *книгой по SRE*).

Вместо этого я хочу рассказать о том, когда впервые начал по-настоящему понимать тему, потому что данная информация должна помочь и вам. Это связано с историей возникновения SRE, описанной Google, поскольку именно тогда Трейнор Слосс изложил свое понимание SRE на первом большом собрании, посвященном данной теме. Я искренне верю, что истории, которые мы сами

---

[1] Настоятельно рекомендую ознакомиться с книгой Алекса Идальго на эту тему: *Hidalgo A.* Implementing Service Level Objectives: A Practical Guide to SLIs, SLOs and Error Budgets. — O'Reilly, 2020.

рассказываем, очень важны для понимания нашей личности, так что это был весьма значимый момент.

Тридцать первого мая 2014 года в городе Санта-Клара, штат Калифорния, я слушал пленарный доклад Трейнора Слосса *Keys to SRE* на самой первой конференции SREcon (https://oreil.ly/cSXef)[1]. Рекомендую и вам посмотреть видеозапись выступления.

Во время того доклада он показал слайд, который подтолкнул меня к пониманию SRE. На рис. 1.1 показан снимок слайда.

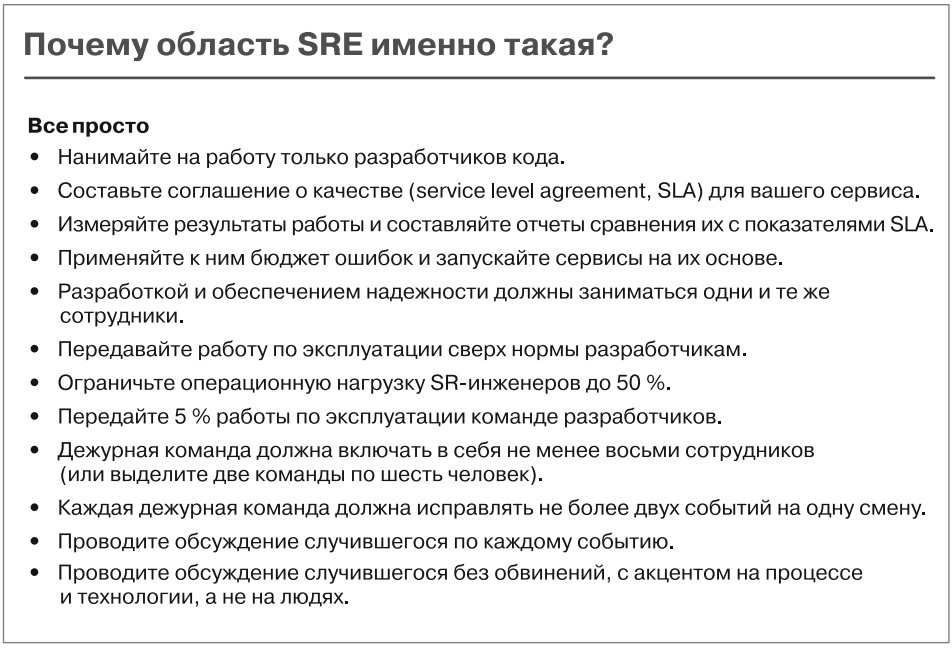

Рис. 1.1. Слайд из пленарного доклада Бена Трейнора Слосса на конференции SREcon14 (в переводе)

На рис. 1.1 представлен список, с которого я начинал, и он по-прежнему отлично подходит для начала работы.

На момент написания книги[2] прошло уже девять лет с того дня, как я впервые увидел данный слайд, и, глядя на него теперь, я поражаюсь тому, сколько пунктов не изменилось с течением времени, и некоторые, похоже, зависят от

[1] Должен признаться: я один из организаторов конференции SREcon.

[2] Конец 2023 года.

контекста работы корпорации Google, в которой они появились. Со времени доклада добавилось несколько нюансов (можно сказать, по меньшей мере три книги; см. ресурсы на тему SRE в приложении B), и, возможно, это тоже веская причина, по которой вы читаете мою книгу.

Связь SRE с разработкой и эксплуатацией

Всякий раз, общаясь с людьми, которые пытаются понять, что такое обеспечение надежности систем, я практически могу гарантировать, что в какой-то момент беседа затронет вопросы вроде: как соотносятся с SRE разработка и эксплуатация (development operations, DevOps)? Какова взаимосвязь между ними? Разумно ли иметь оба отдела в одной компании? Это непростые вопросы, и у меня ушли годы на поиск убедительных ответов на них. Именно поэтому главу 12 в книге *Seeking SRE*, посвященную этой теме, я решил сделать коллективной. На тот момент у меня не было исчерпывающего ответа, и я очень надеялся, что его даст кто-нибудь другой[1].

С помощью ответов из той главы, а также в результате последующей переоценки ценностей и проведенного исследования я наконец нашел решение, которое мне понравилось. Осознав, что для ответа на эти вопросы потребуется применить несколько подходов, я сумел сформулировать многокомпонентное определение, которое меня устроило. Надеюсь, оно устроит и вас.

Позвольте изложить все три части с небольшими комментариями к каждой.

Часть 1. В области SRE задействованы высококлассные DevOps-специалисты

Идея заимствована из главы 1 книги *Site Reliability Workbook* и последующих сообщений корпорации Google. Если вы не программист, но читаете эту книгу, хочу пояснить, что SRE — одна из реализаций[2] общей философии DevOps. Это не самое мое любимое сравнение по нескольким причинам.

- Не будучи специалистом в области программирования, невозможно понять формулировку и ее нюансы.

[1] Один из моих любимых ответов дал Майкл Доэрти, который сказал: «Обеспечение надежности систем: мы не знаем, что такое DevOps, но знаем, что занимаемся немного другими вещами». Этот ответ не входит в официальные, приведенные выше, но я не могу с ним спорить.

[2] В частности, она носит директивный характер, поскольку сфера DevOps во многом старается избегать ограничений, накладываемых конкретной методологией или инструментами. Удалось ли этого добиться — еще одна интересная тема для беседы.

- Мне кажется, я не знаю другой реализации DevOps, кроме стандартной, которая развивалась в течение долгого времени и применялась на практике в реальной жизни, так что идея кажется немного сомнительной.
- В сравнении содержится намек на историческую связь, уходящую корнями в туманные времена, к истокам SRE (или, по крайней мере, к открытию двух направлений), но я не нашел доказательств в его поддержку.
- Я все еще не уверен, что согласен с этим сравнением.

Я не отказываюсь от данной идеи о SRE и DevOps, потому что (помимо того, что она исходит от людей умнее меня) она действительно отражает сходство или, по крайней мере, места пересечения, общие для двух современных методологий создания ПО.

Часть 2. Область SRE ассоциируется с надежностью, а DevOps — с доставкой

Не знаю, как у вас, а у меня периодически случается кризис веры в DevOps. В этот раз я понял, что не могу найти описание области DevOps или принятых в ней методов, которое помогло бы мне сразу выделить ее среди других подходов эксплуатации, существующих в мире. Мне нужны были слова, которые бы позволили сразу отличить DevOps от всего остального, чтобы можно было однозначно выбрать ее из списка («Пункт 3, это же DevOps! Я бы узнал их где угодно!»). Я просмотрел все свои ресурсы и лично составленный список аббревиатур из области DevOps, но безуспешно.

В случае со SRE я могу сказать: «SRE — это надежность». Если меня спросят «Какие методы эксплуатации уделяют особое внимание надежности?», то простой ответ будет — SRE. Поэтому я решил спросить у светил в области DevOps: «Если SRE — это надежность, как одним словом охарактеризовать DevOps?»

Я обращался со своим вопросом к одному светилу, потом к другому (все они были очень милы), пока наконец не получил от Донована Брауна ответ, который меня порадовал. Для него суть DevOps — в *доставке*. Доставке ценности клиентам, доставке программного обеспечения и т. д. В итоге я нашел слово, которое искал.

Область SRE ассоциируется с надежностью, а DevOps — с доставкой.

Я могу с этим жить.

Часть 3. Все дело в направлении внимания

Этот последний фрагмент головоломки я получил от своего друга Тома Лимончелли, который был достаточно любезен и представил его в виде ответа на мою просьбу прислать материалы для коллективной главы книги *Seeking SRE*

(я упоминал о ней ранее). Рисунок 1.2 — иллюстрация к этой главе (изменена по его просьбе).

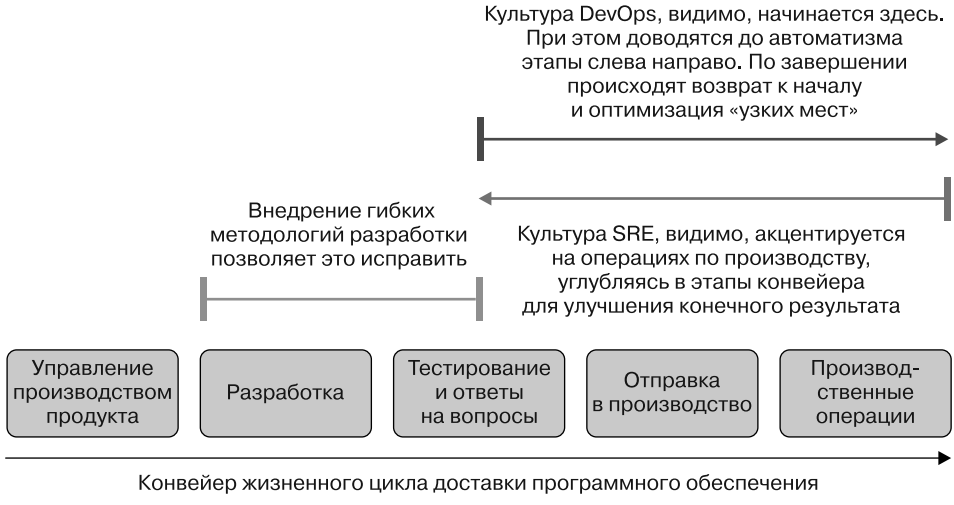

Рис. 1.2. Модель Лимончелли для стратегий SRE, DevOps и Agile. Изменена по сравнению с оригиналом, приведенным в книге Seeking SRE (в переводе)

Кое в чем эта модель нравится мне больше других, поскольку в ней поясняются возникающие на практике пересечения между областями DevOps и SRE, которые, на первый взгляд, не прослеживаются в их подходах или целях. Чуть позже я приведу примеры, а ниже представлено мое лучшее резюме теории Тома.

1. История DevOps начинается с того, что разработчик набирает код на ноутбуке. DevOps-инженер занимается (среди прочего) работой, которую необходимо выполнить для доставки кода в производство (production), чтобы клиенты могли пользоваться им с максимальной выгодой. Внимание направлено от ноутбука к производству[1]. Можно предположить, что это одна из причин того, почему системы непрерывной интеграции и непрерывной доставки (continuous integration and continuous delivery, CI/CD) занимают такое значимое место в арсенале инструментов DevOps-инженеров, их наборе навыков и указаны в объявлениях о вакансиях.
2. SRE начинается с другого. Оно начинается с производства (и действительно, работа SR-инженеров сосредоточена именно здесь). Что должен сделать SR-инженер, чтобы создать надежную среду производства? Для ответа на

[1] С остановками по пути для сохранения кода в репозитории и тестирования с целью убедиться, что его можно безопасно развернуть, чтобы клиенты могли начать пользоваться им.

этот вопрос потребуется посмотреть назад на этапы, предшествующие производству, повторяя вопрос на каждом, пока не доберетесь до ноутбука разработчика.

3. Разное направление внимания. Могут применяться одни и те же инструменты (например, конвейер CI/CD), но с разными целями. В сферах DevOps и SRE можно активно заниматься созданием системы мониторинга, но делать это по разным причинам[1].

И здесь мы подходим к ответу на поставленный выше вопрос: могут ли/должны ли SR- и DevOps-инженеры сосуществовать в одной организации? Для меня ответ — да[2]. Возможно, некоторые их инструменты, а иногда и навыки будут совпадать, но они акцентируются на разных вещах и обеспечивают организации разные преимущества.

Переходим к основам SRE

Теперь, если кто-то спросит: «Что же такое SRE?» — у вас есть базовые знания, чтобы объяснить им. Я расскажу об этом гораздо подробнее в главе 4. А сейчас, когда мы немного обсудили определения и историю SRE, перейдем действительно к основам SRE, которые станут ключевыми для понимания темы и остальных частей книги.

---

[1] Мне не встречались подобные исследования, но интуиция подсказывает, что в результате они будут отслеживать разные вещи. Было бы интересно изучить этот аспект.

[2] При наличии подходящих условий, таких как размер организации (новому стартапу, возможно, не нужны оба специалиста), культура компании (если SR-инженер в нее вписывается, о чем мы поговорим далее) и потребности (нанимайте тех специалистов, которые нужны, а не коллекционируйте всех подряд).