Оглавление

3 Управление рисками

Работа по сопровождению и эксплуатации — это набор обязательств[1] и действия, направленные на их выполнение. В главе 2 мы обсуждали, как создавать и формулировать эти обязательства, контролировать и документировать их выполнение. Управление рисками — это то, что мы делаем для выявления, оценки и приоритизации факторов неопределенности, которые могут привести к нарушению наших обязательств. Это также касается применения ресурсов (технологий, инструментов, людей и процессов) для мониторинга и снижения вероятности возникновения неопределенностей.

Это не наука о достижении идеала! Цель здесь не в том, чтобы устранить все риски. Это была бы безумная цель, которая лишь зря поглощала бы ресурсы. Цель состоит в том, чтобы внедрить оценку и снижение рисков во все наши процессы и постепенно уменьшить воздействие рисков, используя различные методы их предотвращения и смягчения. Данный процесс должен идти постоянно с учетом наблюдений за инцидентами, внедрения новых архитектурных компонентов, увеличения или уменьшения эффекта по мере развития организации. Цикл процесса можно разбить на семь категорий:

- определить возможные опасности/угрозы, создающие операционный риск для сервиса;
- оценить каждый риск с учетом его вероятности и последствий;
- классифицировать вероятность и последствия рисков;
- определить средства управления для смягчения последствий или снижения вероятности риска;
- определить приоритетность рисков, с которыми нужно бороться в первую очередь;
- внедрить средства управления и контролировать эффективность;
- повторить процесс.

---

[1] Выраженных в SLO. — *Примеч. науч. ред.*

Повторяя этот процесс, вы практикуете *кайдзен*, то есть постоянно совершенствуетесь. И самое важное здесь — оценка рисков, где нужно развивать стратегию постепенно.

Факторы риска

Качество *процессов оценки рисков* зависит от множества факторов. Их можно разбить на следующие категории:

- неизвестные факторы и сложность;
- наличие ресурсов;
- человеческий фактор;
- групповые факторы.

Чтобы разработать реалистичный процесс для вашей команды, необходимо учесть все эти факторы, и мы кратко рассмотрим в данной главе каждый из них.

Неизвестные факторы и сложность

Трудность оценки рисков заключается в огромной сложности современных систем. Чем сложнее и запутаннее предметная область, тем больше трудностей испытывают люди, перенося свои знания на ситуации, с которыми еще не сталкивались. Тенденция к чрезмерному упрощению понятий, чтобы с ними можно было легко справиться, называется *редуктивным смещением*. То, что работает при начальном обучении, не работает, когда приобретаются более глубокие знания. Существует огромное количество неизвестных рисков, многие из которых находятся вне нашего контроля. Вот некоторые примеры таких рисков:

- влияние других клиентов внешних совместно используемых систем, таких как Amazon или Google;
- влияние поставщиков, интегрированных в инфраструктуру;
- обновление кода программистами;
- движения на рынке, создающие всплески нагрузки на систему;
- выше- и нижестоящие сервисы в иерархии;
- патчи, изменения в репозитории и другие постепенно накапливающиеся изменения программного обеспечения.

Решение проблем в этих областях очень помогает оценивать риски в таких средах. Служба эксплуатации должна использовать свой коллективный опыт и постоянно

наращивать знания, чтобы строить все более точные модели планирования. Необходимо также понимать, что учесть все заранее невозможно, поэтому следует создавать систему отказоустойчивой и адаптируемой к новым неизвестным обстоятельствам.

Наличие ресурсов

Те из нас, кому приходилось работать в отделе, которому не выделяют достаточно средств, или в бессистемном стартапе, знают, что получить ресурсы для длительных, работающих на перспективу задач, подобных рассматриваемой, бывает... скажем так, непросто (слышали про сизифов труд?). Например, у вас может быть 4 часа или даже всего 30 минут в месяц на то, чтобы заняться процессами управления рисками. Поэтому ваша работа должна давать полезный результат — «приносить прибыль». Цена ваших времени и ресурсов, используемых для смягчения рисков, должна быть меньше цены бездействия. Другими словами, в первую очередь уделяйте время самым вероятным рискам, а также тем, последствия которых могут оказаться наиболее серьезными. Создавайте устойчивые системы и делайте выводы из возникающих инцидентов.

Человеческий фактор

Стоит людям начать что-то делать, как возникает *множество потенциальных проблем* (http://bit.ly/2zyoBmm). Мы все великолепны, но в прилагающихся к нам инструкциях много написанного мелким шрифтом. Вот некоторые моменты, которые могут испортить дело.

- *Синдром бездействия*. Многие сотрудники эксплуатационных отделов обнаружат, что их начальник или коллеги не склонны к риску. Будучи инертными по природе, эти люди выбирают бездействие, поскольку считают, что от перемен риск выше, чем от бездействия. Важно уметь просчитывать риск, а не опускать руки перед неизвестностью.
- *Игнорирование знакомых опасностей*. Опытные инженеры часто игнорируют типичные риски, уделяя больше внимания экзотическим и редким событиям. Например, те, кто привык иметь дело с переполнением дисков, могут сосредоточиться на проблемах центра обработки данных в целом, забыв адекватно спланировать управление дисковым пространством.
- *Страх*. Страх как фактор стресса может иметь как положительный, так и отрицательный эффект. Некоторые люди отлично чувствуют себя в условиях большой нагрузки и высоких ставок — они будут очень полезны, работая в области планирования, смягчения рисков и других вопросов промышленной эксплуатации систем. Но если человек действительно боится, он из-за этого нередко игнорирует

худшие сценарии. Это может привести к недостаточной проработке ключевых, наиболее ответственных и подверженных рискам компонентов и систем. Поэтому важно распознать такие реакции, если они есть у членов вашей команды.

- *Излишний оптимизм*. Другая тенденция человеческого поведения как реакции на риск — излишний и необоснованный оптимизм. Мы часто верим в лучшее относительно себя и других членов наших команд и поэтому склонны рассматривать только идеальные ситуации (никто не утомлен, нас не отвлекают другие инциденты, младшие сотрудники принимают участие в работе). Это касается не только людей, но и происходящих событий. Приходилось ли вам думать: *«Три диска в один день из строя не выйдут»*, а потом получить некачественную партию дисков, с которой именно это и случится?

Обсуждая риски, мы также должны учитывать такие факторы, как утомляемость и отвлечение на выполняемые вручную восстановительные работы (так сказать, аврал-пожар). Каждый раз, оценивая трудоемкость и риски, например, при внесении изменений вручную или при исследовании инцидента, мы должны учитывать, что специалисты-эксплуатационщики, погружающиеся в новую сложную проблему, пришли к нам, возможно, после долгого трудового дня. Может быть, это и не так, но это тоже нужно учитывать. При разработке средств управления для уменьшения или устранения рисков также следует учитывать, что тот, кто будет выполнять ручное восстановление, тоже может быть утомлен или, возможно, тушит несколько пожаров одновременно.

Усталость от оповещений

Усталость от оповещений (http://bit.ly/2zyfqCv) — это ситуация, когда перегрузка и усталость возникают из-за слишком большого количества ненужных сообщений. Вы должны учитывать это, принимая решение о том, сколько предупреждений (для реагирования и исправления вручную) встроено в процессы мониторинга. Такая ситуация бывает вызвана ложными срабатываниями (оповещениями о проблемах, которые на самом деле проблемами не являются, часто из-за плохо настроенных пороговых значений) или тем, что оповещения используются вместо предупреждений о тенденциях, способных стать опасными в ближайшем будущем.

Групповые факторы

Подобно тому как у каждого человека есть свои «слепые зоны», группам тоже свойственны особенности поведения и тенденции, которые могут нарушать процесс управления рисками. Вот факторы, которые следует учитывать.

- *Поляризация группы*. Поляризация группы, или «уклон в риск», возникает потому, что группы, как правило, принимают более экстремальные решения, чем

их отдельные члены. Это приводит к тенденции сдвига в противоположную от первоначальных взглядов сторону. Например, люди, обычно осторожные и осмотрительные, среди единомышленников будут более склонны к риску. А те, кто относился к риску спокойно, станут стремиться его избежать. Люди часто не хотят выглядеть наиболее консервативными в группе. Это может подтолкнуть команду рисковать сверх необходимого.

- *Передача риска*. Группы также будут допускать больше риска, когда у них есть тот, на кого можно его переложить. Например, планируя работу группы эксплуатации, мы можем пойти на больший риск, если знаем, что у нас есть отдельная группа сопровождения базы данных, к которой можно обратиться. Здесь помогут выработка чувства сопричастности и кросс-функциональные команды, где перекладывать риск не на кого.
- *Передача решения*. Передача решения может произойти в том случае, когда команды переоценивают риск и поэтому стремятся передать ответственность за конкретные решения другим. Например, если изменения, риск которых оценивается как высокий, требуют одобрения технического директора (следовательно, и ответственность ляжет тоже на него), то люди будут склонны завышать оценки риска, чтобы спихнуть принятие решений с себя вверх по цепочке. Этого можно избежать, создавая более автономные группы, которые опираются на знания и опыт отдельных специалистов и групп, а не на процедуры иерархического утверждения.

Что мы делаем

Реальность такова, что процесс управления рисками легко может стать чересчур обременительным. Даже при наличии значительных ресурсов команды все равно не смогут охватить все потенциальные риски, способные повлиять на доступность, производительность, стабильность и безопасность системы. Имеет смысл строить процесс повторяющимся и совершенствующимся с течением времени. Кроме того, предпочтение устойчивости в управлении рисками вместо стремления полностью их устранить позволяет разумно рисковать во имя инноваций и улучшений.

Мы хотели бы подчеркнуть, что устранение всех рисков — плохая цель. У систем, не испытывающих стрессовых воздействий, нет стимулов к улучшению и укреплению. В результате они оказываются нестойкими по отношению к неизвестным, не запланированным заранее факторам. Системы, регулярно испытывающие стрессы и, следовательно, спроектированные с учетом обеспечения устойчивости, способны успешнее справляться с неизвестными рисками.

Есть основания считать, что для систем следует планировать бюджет времени простоя, как в Google, чтобы реализовать новые перспективные возможности, сохраняя контроль над риском. В Google при бюджете простоя 30 минут за квартал, если это время не было израсходовано, считают оправданным допустить повышение риска ради обновления функций, улучшений и усовершенствований. Использовать бюджет простоя для внедрения инноваций, вместо того чтобы полностью уклоняться от риска, — это превосходно.

Итак, как из этого можно сформировать реальный подход к оценке риска как процесса? Начнем с того, чего делать не надо!

Чего не надо делать

Итак, нам нужно многое принять во внимание! Вот несколько советов, которые необходимо учесть при изучении процесса управления рисками — пока еще не поздно:

- не позволяйте субъективным предубеждениям повредить вашему процессу;
- не допускайте, чтобы основным источником оценки риска были анекдоты и сарафанное радио;
- не сосредотачивайтесь только на предыдущих инцидентах и проблемах — смотрите вперед;
- не останавливайтесь, сделанное ранее можно и нужно пересматривать;
- не игнорируйте человеческий фактор;
- не игнорируйте эволюцию архитектуры и рабочего процесса;
- не думайте, что ваша среда сейчас такая же, как и прежде;
- не создавайте нестабильные средства управления и не игнорируйте вероятность, что все пойдет не так.

Несомненно, со временем вы дополните этот список, но он неплох, чтобы держать его в памяти для начала, — это поможет избежать подводных камней, ожидающих вас при изучении своих систем.

Рабочий процесс: запуск

Независимо от того, является ли сервис новым или наследником существующего, процесс начинается с запуска. В ходе запуска процесса (рис. 3.1) цель состоит в том,

чтобы распознать основные риски, которые могут поставить под угрозу SLO вашего сервиса, или наиболее вероятные риски.

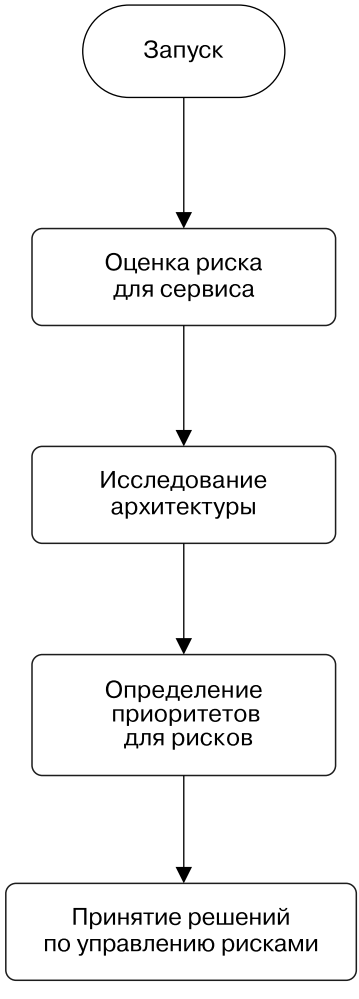

Рис. 3.1. Запуск процесса управления рисками

Кроме того, мы должны учитывать наихудшие сценарии, которые могут угрожать жизнеспособности сервиса в долгосрочной перспективе. Вам — бравому инженеру БД — следует помнить, что мы сейчас не пытаемся охватить все возможные риски. Пока что наша цель — составить начальный список рисков, которые мы будем пытаться смягчить или устранить в первую очередь, и план, позволяющий наиболее полно и с наибольшей отдачей использовать имеющиеся ресурсы.

Оценка риска для сервиса

Вооружившись списком поддерживаемых сервисов и микросервисов, вы должны сесть за стол вместе с владельцами продукта и оценить допустимость рисков для каждого из этих сервисов. Вам необходимо ответить на следующие вопросы.

- Каковы SLO доступности и времени ожидания, определенные для данного сервиса?
- Что представляют собой время простоя или недопустимая задержка для этой услуги, если:
  - затронуты все клиенты;
  - затронута часть клиентов;
  - сервис работает в режиме ограниченной функциональности (только чтение, отключены некоторые функции и т. п.);
  - снижена производительность сервиса?
- Какова цена простоя сервиса:
  - какова потеря дохода;
  - каково влияние на лояльность клиентов:
    - это бесплатный или платный сервис;
    - есть ли конкуренты, к которым клиент может легко перейти;
  - есть ли последствия простоя, способные подорвать работу всей компании:
    - потеря данных;
    - нарушение конфиденциальности;
    - простои во время праздника, иного особого события;
    - большая длительность простоя?

Рассмотрим пример. Компания UberPony специализируется на услуге предоставления клиентам пони по вызову, и ею предлагаются следующие шесть сервисов.

1. Регистрация нового клиента.
2. Прием и выполнение заказа на предоставление «пони по требованию».
3. Регистрация водителя пони.
4. Логистика для водителей пони.
5. Выплаты водителю пони.
6. Внутренняя аналитика.

Рассмотрим сервисы «Регистрация клиентов» (табл. 3.1) и «Заказ и выполнение заказов» (табл. 3.2).

Таблица 3.1. Регистрация клиентов UberPony

SLO доступности	99,90 %
SLO задержки	1 секунда
Количество новых клиентов в день	5000
SLO допустимых ошибок	5
Затраты на инфраструктуру в сутки	13 698 долларов
Затраты на инфраструктуру на 1 доллар дохода	0,003 доллара
Показатель ценности клиента[1]	1000 долларов
Показатель ценности новых клиентов за день (суммарно)	5 000 000 долларов
Пиковое количество обращений клиентов в минуту	100
Отсев клиентов после ошибки	60 %
Пиковое значение потерь за минуту	60 000 долларов

Таблица 3.2. Заказ и выполнение заказов в UberPony

SLO доступности	99,90 %
SLO задержки	1 секунда
Количество текущих заказов в день	500 000
SLO допустимых ошибок	500
Затраты на инфраструктуру в сутки	30 000 долларов
Затраты на инфраструктуру в пересчете на 1 доллар дохода	0,006 доллара
Средний доход от одного заказа	10 долларов
Ежедневный доход	5 000 000 долларов
Пиковое количество заказов в минуту	1000
Отмены заказов после ошибки	25 %
Отсев клиентов после ошибки	1 %
Пиковое значение потерь дохода за минуту	2500 долларов
Снижение суммарного показателя ценности клиентов (перспективные потери от отсева клиентов) за минуту	10 000 долларов
Всего потерь за минуту	12 500 долларов

Таким образом, похоже, что проблемы в сервисе регистрации клиентов могут обойтись нам в 4,8 раза дороже по сравнению с сервисом приема и выполнения заказов. Получается, 75 % клиентов будут повторять попытку заказа после ошиб-

[1] Англ. customer lifetime value — ожидаемая прибыль от одного клиента в течение всего жизненного цикла. — *Примеч. науч. ред.*

ки, но только 40 % вернутся, если не смогут зарегистрироваться. Видимо, вместо этого они с радостью уйдут на UberDonkey. Обратите внимание: мы попытались учесть такие переменные, как потеря клиента после ошибки заказа и количество повторных попыток клиентов зарегистрироваться или сделать заказ после ошибки. Это будет непросто без хорошей бизнес-аналитики, но при недостатке данных можно довольствоваться хотя бы приближенными оценками. Во всяком случае, это лучше, чем ничего!

Со временем данные будут изменяться и дополняться, поэтому обязательно вовремя обновляйте показатели. Например, если UberDonkey станет более конкурентоспособным и UberPony начнет терять 5 % клиентов из-за ошибки в заказе, то наши потери за минуту простоя сервиса приема и выполнения заказов внезапно вырастут до 52 500 долларов, что значительно повысит его приоритет. Тогда нам будет важнее переключить внимание на этот сервис.

Ревизия архитектуры

Теперь, определив сферу деятельности, проведем ревизию систем и сред, входящих в сферу нашей ответственность:

- центры обработки данных (ЦОД);
- архитектурные компоненты/уровни (СУБД MySQL, балансировщики нагрузки Nginx, экземпляры приложений J2EE, сеть, брандмауэр, Hadoop/HDFS, сеть доставки контента (CDN));
- распределение ролей внутри этого компонента (Writer/Primary, Replica);
- пути и порядок взаимодействия между сервисами (запросы из приложения в MySQL, из балансировщика нагрузки в приложение, публикация приложения в Redis);
- задания (извлечение, преобразование и загрузка (ETL), загрузка CDN, обновление кэша, управление конфигурацией, оркестрация (общая координация), резервное копирование и восстановление, агрегация журналов).

В табл. 3.3 представлен упрощенный перечень первоочередных сервисов.

Таблица 3.3. Регистрация клиентов UberPony

Компонент	**ЦОД 1**	**ЦОД 2**
Фронтальные балансировщики нагрузки	2	2
Веб-серверы	20	20
Балансировщики нагрузки Java	2	2
Java-серверы	10	10

Продолжение ⇘

Таблица 3.3 *(продолжение)*

Компонент	ЦОД 1	ЦОД 2
Прокси базы данных	2	2
CloudFront CDN	Сервис	Сервис
Кэш-серверы Redis	4	4
Серверы записи кластера MySQL	1	0
Серверы чтения кластера MySQL	2	2
Репликация MySQL	Сервис	Сервис
Обновление CDN	Задание	Задание
Обновление кэша Redis	Задание	Задание
Резервные копии MySQL	Задание	Не определено
Процесс ETL	Задание	Не определено
Хранилище данных RedShift	Сервис	Не определено

Наш следующий шаг — оценить риски в этой архитектуре, которые могут повлиять на работу сервиса.

Расстановка приоритетов

Как выявить риски, способные привести к нарушению целей SLO, и расставить их приоритеты? В области управления рисками используется определение риска как произведение вероятности[1] возникновения фактора, приводящего к неблагоприятному исходу, и величины последствий этого исхода.

Пример спектра возможных оценок приведен в табл. 3.4.

Таблица 3.4. Спектр оценки

Вероятность/ последствия	Тяжелые	Значительные	Умеренные	Незначительные	Ничтожно малые
Почти наверняка	Неприемлемый	Неприемлемый	Высокий	Средний	Приемлемый
Очень вероятно	Неприемлемый	Высокий	Высокий	Средний	Приемлемый
Возможно	Неприемлемый	Высокий	Средний	Средний	Приемлемый
Маловероятно	Высокий	Средний	Средний	Приемлемый	Приемлемый
Редко	Высокий	Средний	Приемлемый	Приемлемый	Приемлемый

[1] В оригинале здесь likelihood; ниже имеется замечание о сложности толкования понятий likelihood и probability. — *Примеч. науч. ред.*

Для устранения двусмысленности важно определить возможные значения для вероятности и цены исхода. Цена может различаться в зависимости от конкретной предметной области и задачи. Что касается неоднозначности понятий вероятности, ожидаемой вероятности, частоты, то мы предлагаем вам изучить публикацию *Describing probability: The limitations of natural language* («Описание вероятности: ограничения естественного языка», http://www.risk-doctor.com/pdf-files/emeamay05.pdf).

Разделим все вероятности следующим образом (табл. 3.5).

Таблица 3.5. Вероятности

Качественная оценка	Количественная оценка, %
Почти наверняка	> 50
Очень вероятно	26–50
Возможно	11–25
Маловероятно	5–10
Редко	< 5

Будем считать эти проценты вероятностью нарушения SLO в течение определенного периода времени, например за неделю. Что же касается последствий, то классифицируем их, оценивая влияние на наши SLO в случае реализации нарушающих нормальный процесс факторов, включая повреждение данных, утрату конфиденциальности и нарушения системы безопасности. Большинство этих проблем попадают в категории сильного или значительного воздействия. Опять же это только примеры.

Тяжелые последствия (немедленное нарушение SLO)

Тяжелые последствия означают следующее.

- Недоступность всего сервиса или ограничение его функциональности на срок в общей сложности от 100 миллисекунд до 10 минут и более для 5 % пользователей и более. (В семидневной неделе 10 080 минут. Десять минут простоя поглощают 99,9 % SLO.)
- Состоявшееся или неизбежное в будущем раскрытие данных клиента другим клиентам.
- Предоставление посторонним лицам доступа к рабочим (production) системам и/или реальным данным.
- Повреждение данных транзакций.

Любое из перечисленных событий переводит последствия в категорию тяжелых.

Значительные последствия (неминуемое нарушение SLO)

Значительные последствия означают следующее.

- Недоступность всего сервиса или ограничение его функциональности на срок от 100 миллисекунд до 3–5 минут для 5 % пользователей и более (до 50 % бюджета времени доступности).
- Уровень доступных мощностей системы снизился до 100 % от необходимого вместо запланированного значения 200 %.

Любое из перечисленных событий переводит последствия в категорию значительных.

Умеренные последствия (возможно нарушение SLO в сочетании с другими инцидентами, произошедшими в тот же период)

Умеренные последствия означают следующее.

- Недоступность всего сервиса или ограничение его функциональности на срок от 100 миллисекунд до 1–3 минут для 5 % пользователей и более (до 33 % бюджета времени доступности).
- Уровень доступных мощностей системы снизился до 125 % от необходимого вместо запланированного значения 200 %.

Любое из перечисленных событий переводит последствия в категорию умеренных.

Незначительные последствия

Незначительные последствия означают следующее.

- Недоступность сервиса или ограничение его функциональности на срок от 100 миллисекунд до 1 минуты для 5 % пользователей и более (до 10 % бюджета времени доступности).
- Уровень доступных мощностей системы снизился до 150 % от необходимого вместо запланированных 200 %.

Любое из перечисленных событий переводит последствия в категорию незначительных.

Напомним: мы не станем пытаться охватить все потенциальные риски. Вы будете пополнять их портфель изо дня в день в рамках текущих процессов управления инцидентами и рисками. То, что мы сейчас делаем, называется *фреймингом*: мы ограничиваем нашу работу некими разумными рамками. В данном случае мы

включаем в эти рамки сценарии, *наиболее вероятные* и имеющие *наибольшие последствия*.

Например, мы знаем, что сбои компонентов и сбои экземпляров — это рядовые события в общедоступных облачных средах, наподобие той, которую UberPony использует в качестве хоста. Другими словами, там малое среднее время между отказами (MTBF). Мы будем классифицировать эти сбои как вероятные для групп веб-экземпляров и групп экземпляров Java, поскольку они будут встречаться нам в умеренном количестве в любое время (от 20 и 10 соответственно). Однако отказ одного веб-экземпляра означает, что будут затронуты 5 % клиентов. При сбое экземпляра Java окажутся затронуты 10 % клиентов. Это является нарушением SLO, и поскольку запуск новой копии данного экземпляра может занять от 3 до 5 минут, это повлечет *значительные* последствия для сервиса. При вероятности, оцениваемой как *«очень вероятно»*, и *значительных* последствиях риск *высокий*. После того как мы включим автоматическое восстановление после сбоев (вывод аварийного экземпляра из эксплуатации и запуск вместо него нового), этот процесс будет занимать, согласно проведенным тестам, в среднем 5 секунд. Это изменит последствия на *незначительные*, а риск соответственно на *умеренный*.

Если в нашем исследовании рассматривать сбой на уровне сервиса или экземпляра, то можно составить примерно такую таблицу (табл. 3.6).

Таблица 3.6. Сервис регистрации клиентов UberPony

Компонент	**Вероятность**	**Последствия**	**Риск**
Балансировщик клиентской нагрузки	Возможно	Тяжелые	Неприемлемый
Веб-серверы	Вероятно	Значительные	Высокий
Балансировщики нагрузки Java	Возможно	Значительные	Высокий
Java-серверы	Вероятно	Значительные	Высокий
Прокси базы данных	Возможно	Значительные	Высокий
Cloudfront CDN	Редко	Значительные	Умеренный
Кэш-серверы Redis	Возможно	Значительные	Умеренный
Серверы записи MySQL	Маловероятно	Тяжелые	Высокий
Серверы чтения MySQL	Возможно	Значительные	Высокий
MySQL-репликация	Возможно	Значительные	Высокий
Обновление CDN	Маловероятно	Незначительные	Приемлемый
Redis-обновление кэша	Маловероятно	Незначительные	Приемлемый
Резервные копии MySQL	Маловероятно	Значительные	Приемлемый
Процесс ETL	Маловероятно	Незначительные	Приемлемый
Хранилище данных RedShift	Редко	Незначительные	Приемлемый

Используя фрейминг, мы сначала погрузимся в ситуации неприемлемого или высокого риска, затем можно заняться случаями умеренного риска и т. д. В разделе, посвященном базам данных, мы выполним более детальную оценку рисков для БД. Наша цель — помочь вам разобраться в процессе. Еще одна особенность заключается в том, что необходимо учитывать также все риски на уровне центра обработки данных. Они встречаются редко, но относятся к той же категории, что и нарушение конфиденциальности, потеря данных и другие риски, требующие рассмотрения в связи с потенциальными последствиями для бизнеса.

Управление рисками и принятие решений

Теперь, когда у нас есть список рисков для оценки с расставленными приоритетами, рассмотрим методы принятия решений по управлению рисками для их смягчения и по возможности устранения. Мы уже начали делать это в предыдущем разделе для веб-серверов и серверов Java, автоматизировав замену аварийных компонентов, чтобы сократить среднее время восстановления (MTTR) после сбоя. Помните, что мы должны позаботиться в первую очередь о быстром восстановлении и сокращении MTTR, а не об устранении сбоев. Наша цель — устойчивость, а не нестабильно высокая доступность!

Почему MTTR важнее, чем MTBF

Создавая систему, которая редко ломается, вы делаете ее изначально неустойчивой. Будет ли ваша команда готова выполнить ремонт, когда система выйдет из строя? Будет ли она вообще знать, что делать? Если же в системе часто происходят отказы, последствия которых смягчаются и остаются незначительными, то команда знает, что делать, когда все идет не так, как надо. Процессы хорошо документированы и отточены, и автоматическое восстановление становится полезным и действует по-настоящему, а не скрывается в темных углах вашей системы.

Для каждого потенциального риска команда может выбрать один из трех подходов:

- избегание (найти способ устранить риск);
- сокращение (найти способ уменьшить последствия риска, когда он реализуется);
- принятие (обозначить риск как допустимый и соответствующим образом запланировать ответные действия).

В сфере управления рисками существует и четвертый подход, называемый разделением рисков, предусматривающий аутсорсинг, страхование и другие способы передачи или делегирования рисков. Однако все это не относится к риску в ИТ, поэтому анализировать его не будем.

Для каждого компонента мы рассмотрим типы сбоев, их последствия и несколько приемов управления, направленных на автоматизацию восстановления, сокращение времени восстановления и снижение частоты сбоев. Это управление потребует определенных затрат и усилий. Сравнивая затраты с возможными потерями при простоях, мы сможем принять правильное решение о выборе мер по снижению рисков.

Выявление рисков

Оценивая риски для компании UberPony, мы определили риски на уровне сервиса хранения данных MySQL как высокие. Это очень типично для уровня базы данных. Итак, посмотрим, что можно сделать, чтобы уменьшить эти риски. В этом сервисе выявлены четыре ключевые точки возможных отказов:

- сбой модуля записи данных;
- сбой модуля чтения данных;
- сбой модуля репликации;
- сбой модуля резервного копирования.

Все это типичные точки отказа для хранилищ данных.

Оценка

Чтобы оценить риск ошибок записи, команда эксплуатации UberPony садится и определяет свои возможности по автоматизации восстановления после сбоя в модуле записи в MySQL. При аварии модуля записи наш сервис «Регистрация клиентов» не может создавать или изменять какие-либо данные. Это означает: никаких новых клиентов и никакой возможности изменять данные ни для самих клиентов, ни для UberPony. Мы уже определили, что неработоспособность сервиса регистрации на пике нагрузки может приводить к потере до 60 000 долларов «стоимости» новых клиентов каждую минуту. Понимать это очень важно! Это означает, что *принятие риска* в данном случае не подходит.

Смягчение рисков и управление ими

Некоторые меры по *устранению риска* у нас уже приняты. У нас есть RAID-массив с десятью дисками, который обеспечивает избыточность хранения данных, поэтому сбой отдельного диска не приводит к сбою всей базы данных. Аналогично избыточность предусмотрена и в среде выполнения. Другой подход к *устранению риска* — замена базового ядра СУБД MySQL на Galera — архитектуру, позволяющую выполнять запись в любой узел кластера MySQL. Это потребует значительных изменений архитектуры, и никто из команды не имеет соответствующего

опыта. Обсудив риски, вносимые новой системой, мы пришли к выводу, что они перевешивают выгоды от нового подхода.

Если приложение спроектировано правильно, то даже при аварии модуля записи клиенты все равно могли бы входить в сервис и просматривать свои данные через модуль чтения. Это и есть *смягчение риска*. Пообщавшись с командой разработчиков программного обеспечения, мы узнаем, что у них это предусмотрено. Но в таком режиме ограниченной функциональности новые клиенты регистрироваться по-прежнему не могут, поэтому ценность этой функции оказывается не очень высокой (однако другие функции, которые пока не разрабатываются, высоко ценятся на конкурентном рынке).

В итоге команда решает реализовать автоматическое восстановление — в данном случае автоматическое переключение на другой мастер-диск. Было выбрано восстановление в автоматическом, а не ручном режиме, потому что за десять минут простоя, допускаемых согласно SLO, оператор просто не успевает войти в систему и выполнить необходимые действия. И поскольку вероятность потери данных при записи все же остается, этот процесс должен быть очень надежным.

Реализация

В качестве технологии для автоматического восстановления при сбоях команда выбрала MySQL MHA. MySQL MHA (MySQL High Availability) — это программное обеспечение для управления восстановлением, переключением аварийных компонентов и соответствующими изменениями в топологии репликации. Прежде чем реализовать столь важный процесс, команда разработала план интенсивного тестирования. Такие тесты выполняются поэтапно, сначала в тестовой среде без трафика, затем в тестовой среде с моделируемым трафиком и, наконец, в реальной производственной среде, сопровождаясь тщательным мониторингом. Эти тесты проводят много раз, чтобы убедиться, что они охватывают не только отдельные частные случаи. Тесты включают в себя:

- корректное отключение основной базы данных в тестовой среде;
- принудительное отключение процесса MySQL в тестовой среде;
- принудительное отключение экземпляра сервера, на котором работает MySQL, в тестовой среде;
- моделирование фрагментации сети на изолированные сегменты.

После каждого теста команда должна:

- записать время, которое потребовалось на восстановление работоспособности;
- записать величину задержки для моделируемого и реального трафика с целью оценки влияния на производительность;

- убедиться в отсутствии повреждений таблиц;
- убедиться в отсутствии потерь данных;
- проверить журналы ошибок на стороне клиентов, чтобы увидеть, как это повлияло на них.

После того как команда убедилась, что система будет работать в соответствии с заданными SLO, она начинает думать, как включить этот процесс в другие ежедневные процессы, чтобы обеспечить его регулярное выполнение, качественное документирование и отсутствие ошибок. Первоначально его решают включить в процесс развертывания, используя процесс восстановления после отказа для выполнения внесения изменений в базы данных, не влияя на однопоточные процессы репликации MySQL. К моменту завершения этих работ время восстановления после отказа сократилось до 30 секунд и менее.

Поскольку процессы аварийного восстановления актуальны и для команды разработки программного обеспечения, они тоже понимают, что в течение этого 30-секундного интервала также возможна потеря данных. Поэтому разработчики выполняют двойную запись для своих приложений, отправляя все вставки, обновления и удаления брокеру событий на случай необходимости восстановления утерянных данных. Это еще одна мера смягчения последствий в случае сбоя модуля записи.

Все эти приемы управления хороши на ранних стадиях построения системы. Важно помнить, что они не обязательно должны быть идеальными. Это лишь самое начало, когда мы решаем в первую очередь наиболее приоритетные и наиболее значимые задачи.

Процесс запуска системы завершен — вы прошли долгий путь, охватив наиболее распространенные случаи рисков. С этого момента начинается постепенное совершенствование.

Постепенное совершенствование

После завершения процесса запуска приоритеты по устранению рисков и смягчению их последствий включаются в общий технологический процесс (architectural pipeline) проектирования, сборки и текущего обслуживания. Ранее мы упоминали, что управление рисками — это непрерывный процесс; таким образом, нет необходимости реализовывать все с самого начала: постепенно развиваясь, процессы дополнят портфель рисков, обеспечивая все более широкий охват (рис. 3.2). Итак, что это за процессы?

- Периодическая ревизия сервисов (service delivery reviews).
- Управление инцидентами.
- Организация технологического процесса.

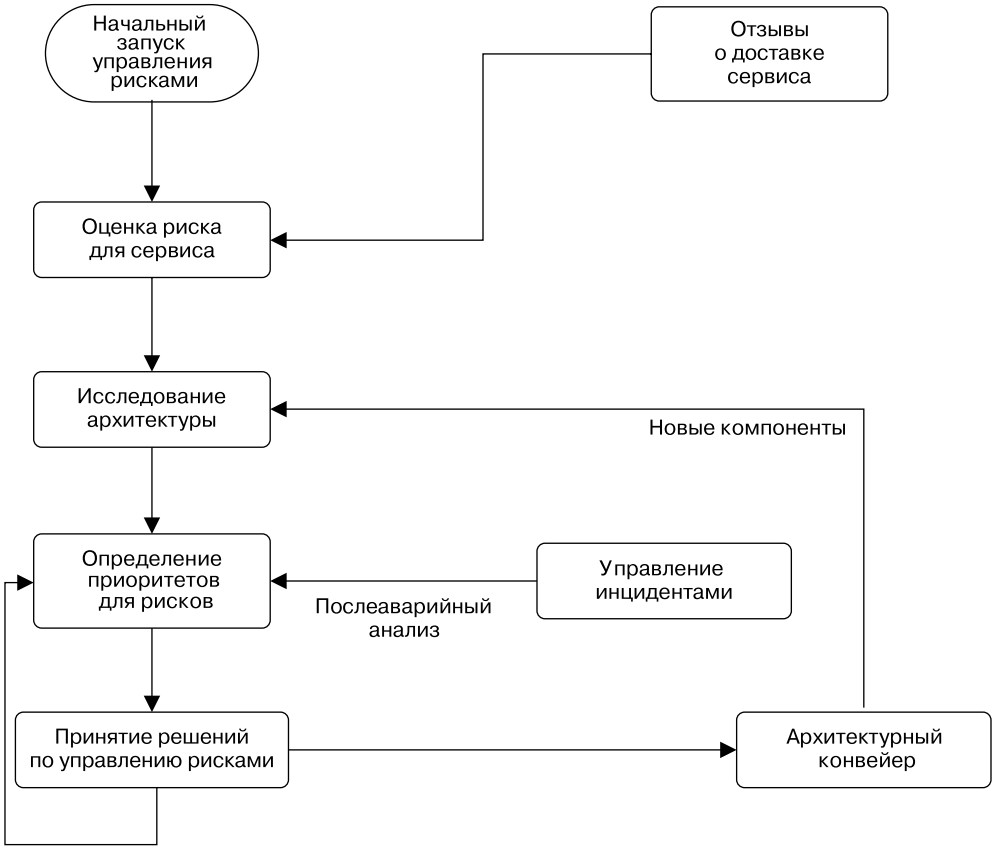

Рис. 3.2. Развивающийся жизненный цикл и его вклад в процесс управления рисками

Ревизия сервисов — это периодический возврат к доработкам сервиса с учетом изменений в допустимости рисков и их последствий, в доходах и пользовательской базе. Если изменения этих параметров существенны, предыдущие классификации рисков и решения для их устранения или смягчения их последствий должны пересматриваться, чтобы оставаться адекватными.

Процессы управления инцидентами также создают предпосылки для изменения приоритетов рисков. По мере выявления новых уязвимостей при анализе аварий необходимо изучать эти уязвимости и добавлять их в список с соответствующими приоритетами. Наконец, в уже построенный технологический процесс необходимо внедрять новые компоненты управления рисками, чтобы выявлять те риски, которые могли быть упущены на этапе проектирования.

Резюме

Итак, вы уже узнали о важности включения управления рисками в повседневные процессы ИТ. Вы ознакомились с некоторыми соображениями и факторами, которые могут повлиять на это, и рассмотрели реалистичный пример процессов как запуска новой системы, так и повседневных, позволяющих поддерживать последовательное развитие системы управления рисками.

Даже понимая наши обязательства в отношении качества уровня обслуживания и потенциальных рисков, угрожающих их выполнению, мы все еще упускаем жизненно важную составляющую — оперативный контроль (operational visibility). Для своевременного выявления проблем и принятия решений о направлении развития наших систем нам необходимо ясное представление о текущей ситуации, равно как знания об изменениях производительности и других характеристик систем с течением времени.

4 Оперативный контроль

Оперативный контроль, часто называемый мониторингом, — это краеугольный камень искусства проектирования надежных баз данных. Оперативный контроль позволяет знать все о рабочих характеристиках сервиса БД благодаря регулярному измерению параметров и сбору сведений о различных компонентах. Почему это так важно? Зачем нужен оперативный контроль? Вот лишь некоторые причины.

- *Исправления и изменения в связи с неисправностями.* Нам нужно знать, когда что-то повреждено или вот-вот будет повреждено, чтобы вовремя это исправить, не допуская нарушения целевых показателей качества обслуживания (SLO).
- *Анализ производительности и поведения.* Важно понимать, каковы задержки в наших приложениях и где они возникают, а также видеть их изменения со временем, включая пиковые значения и всплески. Эти данные играют решающую роль в понимании того, как влияет использование новых функций, эксперименты и оптимизация.
- *Планирование мощностей.* Возможность соотносить поведение пользователя и эффективность приложений с реальными ресурсами — процессором, сетью, хранилищем данных, пропускной способностью и памятью — крайне важна для предотвращения нехватки ресурсов в критический для бизнеса момент.
- *Отладка и послеаварийный анализ.* Чем интенсивнее изменения, тем чаще что-нибудь оказывается повреждено. Хороший оперативный контроль дает возможность быстро находить точки сбоя и точки оптимизации, позволяющие снизить риск в будущем. Человеческие ошибки никогда не были основной причиной поломок, но системы всегда можно улучшить и сделать их более устойчивыми.