1. **Добавьте в документ раздел «Мотивация».**

Мониторинг позволит узнать, а что на самом деле случилось.

Но коллегам скорее нужна, просто осторожность при выводе новых фич в продакшн.

По факту у них есть всего 1 интернет магазин, 1 CRM, 1 MES, 2 API, RabbitMQ, 2 базы PostgreSQL. Здесь опять же мое личное мнение.

Но честно, я бы не рассматривал мониторинг на данном этапе развития компании отдельно, а только как часть решения внедрения «Наблюдаемости». Т.е политическое решение руководства компании, чтобы такое не повторилось никогда…И руководство готово потратить много денег на решение этого вопроса

Проактивный подход, а это уже как раз принцип Observability сэкономит время и деньги, а главное нервы. Но по факту сам Мониторинг — реактивный: ну построим мы дашборды и обложимся алертами, и у нас будет только ответ на вопрос «А что собственно случилось?»

Мониторинг — сбор бизнесовых и инфраструктурных метрик (metrics). Нам сейчас нужны только инфраструктурные, но бюджет проекта мы само собой будем прикрывать бизнесовыми.

Нам нужно - отслеживание доступности, времени отклика, ошибок и простоев.

Мониторинг производительности приложений (Магазин не обязательно, CRM по возможности, MES обязательно+API ) поможет непрерывно и автоматически оценивать работоспособность приложений, не мешая пользователям.

Вот что компания получит при внедрении мониторинга.

1. **Добавьте раздел «Выбор подхода к мониторингу».**

В нашем примере нам нужно минимум 3 вида мониторинга:

* Мониторинг производительности CPU, RAM, диски
* Сетевой мониторинг обработка запросов
* Мониторинг базы данных.

Мы сразу отказываемся от Метода USE, так как он не будет корректно работать при использования кеширования.

Остается 4 сигнала и RED.

Я останавливаюсь на RED из-за возможность реализации распределённой трассировки. Так как она позволит отслеживать запрос в разных сервисах и измерять время, которое заняли события в каждом сервисе (а это позволит смониторить наши потерянные заказы).

1. **Опишите, какие метрики и в каких частях системы вы будете отслеживать.**
2. Number of dead-letter-exchange letters in RabbitMQ (Так как RabbitMQ не имеет свою бд, в которой будут сохранены удаленные сообщения, придется собирать эту метрику).
3. Number of message in flight in RabbitMQ (Так как RabbitMQ по умолчанию пытается сообщения так быстро, как только сможет. А у нас по решению ограничение потока. Данная метрика нужна для сбора данных для Backpressure / Circuit Breaker).
4. Number of requests (RPS) for internet shop API, CRM API, MES API (собираем, так как нам нужно выровняться по равномерно нагрузке по всем API, понятно MES API всегда будет перенагружен).
5. CPU % for shop API (При возможности. Нагрузка через магазин равномерна, +100 в месяц. Ситуация прогнозируема.)
6. CPU % for CRM API (При возможности. Основная нагрузка через очередь MES. Ситуация прогнозируема.)
7. CPU % for MES API (Обязательно. Нагрузка не прогнозируемая.)
8. Memory Utilisation for shop API, CRM API, MES API (MES обязательно, остальное по возможности)
9. Memory Utilisation for shop db instance (Только при увеличении нагрузки через магазин).
10. Memory Utilisation for MES db instance (Обязательно. После внедрения кластера. Все равно нужны показатели)
11. Number of connections for shop db instance (Нет описания принципа взаимодействия, но скорее это всего технических пользователь с настроенными лимитами коннекта в самой базе. В метрике нет необходимости. Только при наличии возможности и времени).
12. Number of connections for MES db instance (Крайне желательно. Хотя в PostgreSQL два простых запроса).
13. Response time (latency) for shop API, CRM API, MES API (для равномерной нагрузки)
14. Size of S3 storage (Обязательно. Если я правильно понимаю, не важно, как создан заказ, модели для расчетов должны храниться в S3. Они не удаляются после выполнения заказа, а должны храниться какой-то срок для разрешения кризисных ситуаций с клиентами – не так посчитали, не так сделали и т.п. )
15. Size of shop db instance(я пока не вижу необходимости. По возможности)
16. Size of MES db instance (Обязательно. У нас там кластер запланирован.)
17. Number of HTTP 200 for shop API, CRM API, MES API (Список успешных сообщений. Нужен. Для построения статистики кол-во успешных / не успешных запросов.)
18. Number of HTTP 500 for CRM API, MES API, shop API (Обязательно. Увеличения этого показателя позволит определить начинающуюся неприятную ситуация. Ну и для статистики)
19. Number of simultaneous sessions for shop API, CRM API , MES API (нужно для ограничения кол-ва одновременного подключения Backpressure / Circuit Breaker. Особенно для MES API).
20. Kb tranferred (received) for shop API (Это же сжатый размер всех переданных ресурсов. Если я правильно понял. Внутренние сервисы. Не нужно.)
21. Kb tranferred (received) for CRM API (Это же сжатый размер всех переданных ресурсов. Если я правильно понял. Внутренние сервисы. Не нужно.)
22. Kb tranferred (received) for MES API (Только если компания будет ограничивать не API не только условиями кол-ва коннектов, но и по кол-ву переданной информации.)
23. Kb provided (sent) for shop API (Внутренние сервисы. Не нужно.)
24. Kb provided (sent) for CRM API (Внутренние сервисы. Не нужно.)
25. Kb provided (sent) for MES API (Только если компания будет ограничивать не API не только условиями кол-ва коннектов, но и по кол-ву переданной информации.)

Я не вижу необходимости в дополнительной детализации метрик, т.е. в использовании ярлыков. Как я уже писал выше не так велика и распределена система, чтобы детализировать метрики.

Предложил бы коллегам «собрать обратную связь, провести исследование по фактическим результатам и определиться с необходимой детализации, поставить в план доработок и т.п.»