Принципы построения высоконагруженных систем
Институт прикладных компьютерных наук ИТМО

Домашнее задание 1. Мониторинг и нагрузочное тестирование

Георгий Семенов
georgii.v.semenov@mail.ru
Мягкий дедлайн: Сб, 22.11.2025, 23:59 МСК
Жесткий дедлайн: Сб, 29.11.2025, 23:59 МСК

Имя Фамилия
DDIA25-HW1-NameSurname.pdf

November 13, 2025

Правила курса

- Задание выдается на 2 недели с двумя дедлайнами:
  - **Мягкий дедлайн** – в рамках этого периода можно заранее отправить решение и получить обратную связь, чтобы исправить замечания до наступления жесткого дедлайна.
  - **Жесткий дедлайн** – после этого новые посылки работы не принимаются, сдать работу больше нельзя.
- Планируется выдать 4 домашних задания, каждое из которых стоит ±12 баллов. Правила выставления оценки за курс следующие:
  - **A** – **«5»** – $\geq 90\%$ от общей суммы обязательных баллов (предв. ≥ 43.2)
  - **B/C** – **«4»** – $\geq 75\%$ от общей суммы обязательных баллов (предв. ≥ 36)
  - **D/E** – **«3»** – $\geq 60\%$ от общей суммы обязательных баллов (предв. ≥ 28.8)
  - **F** – **«2»** – $< 60\%$ от общей суммы обязательных баллов (предв. < 28.8)
- Выполненное задание рекомендуется оформить в LaTeX(например, в Overleaf) и выслать файлом с именем вида `DDIA25-HW1-IvanIvanov.pdf` на почту выше.
  - Пожалуйста, оформляйте ответы на задания в блоке `Решение`.

Задание	1.1	1.2	1.3	1.4 (*)	2.1	2.2	Σ	Σ (*)
Баллы	1.5	2.5	3	2	1.5	3.5	12	14

1 Мониторинг

В ходе выполнения заданий в этом разделе вам предстоит помочь корпорации *Монокль* внедрить актуальные стандарты мониторинга на основе Prometheus.

1.1 Метрики и их классы

Ранее в *Монокле* у каждого бизнес-юнита были свои политики мониторинга и надежности, наиболее подходящие его продуктам и сервисам. Настал момент перейти к централизованной системе мониторинга и внедрить единые стандарты телеметрии, в т.ч. сбора метрик.

Задание: (1.5 б.) Помогите SRE-команде *Монокля* сформировать глоссарий терминов для нового корпоративного стандарта *observability*. Заполните таблицы ниже.

Решение:

«Observability Pillar»	Характеристика

«Golden Signal»	Пример метрики	Единица измерения

Метрика доступности	Описание
Uptime (%)	
Downtime (s)	
MTBF (s)	
MTTR (s)	
MTTD (s)	

□

1.2 SLA

Бизнес-юнит *Монолит* сейчас использует собственную систему мониторинга *Монада* со своим форматом хранения метрик. Метрики *Монады* хранятся в иерархической структуре (например, `//monolith/web/http_requests/timings`) и соответствуют лишь двум типам: RPS-счётчики (`rate`) и перцентили (`percentile`). Поддерживаются только перцентили из множества $(0.5, 0.75, 0.9, 0.95, 0.99)$.

Настало время «распилить *Монолит*» и канонично присоединиться к прогрессивному миру PromQL. SRE-команда *БЮ Монолит* добродушно реализовала PromQL-адаптер к своей системе мониторинга. С ним можно выбрать метрику *Монады* с помощью метки `monad`, указав путь `path`, облачную зону `cloud_dc` (поддерживаются зоны `AB, BC, CA`), тип поля и метрики `field` и `type` (что бы эта идиома ни значила).

Стажер в SRE-команде *Монокля* получил задачу реализовать централизованный мониторинг SLO на основе предоставленного PromQL-адаптера для бэкенда *Монолита* – необходимо отслеживать 99-ый перцентиль времени ответа и суммарный RPS запросов:

```
avg (monad{
        cloud_dc=~".*",
        path="//monolith/web/http_requests/timings",
        field="p99",
        type="quantile"
})

sum (monad{
        cloud_dc=~".*",
        path="//monolith/web/http_requests/requests",
        field="1",
        type="rate"
})
```

Задание: Помогите SRE-команде *Монокля* оценить и обосновать корректность результата их стажера.

1. (0.75 б.) Приведите пример инцидента, при котором метрики стажера не зафиксируют реальную деградацию доступности сервиса для $> 1\%$ пользователей *Монолита*.

2. (0.75 б.) Поможет ли использование взвешенного среднего вместо простого усреднения по зонам избежать проблемы в пункте выше? Проиллюстрируйте ответ.

3. (0.5 б.) Предложите, как можно усовершенствовать предоставленный PromQL-адаптер, чтобы помочь стажеру правильно составить необходимые запросы.

4. (0.5 б.) Запишите корректные PromQL-выражения для 99-го перцентиля времени ответа и суммарного RPS запросов в предположении, что эти метрики изначально были доступны как «настоящие» Prometheus-метрики, а не как адаптер через *label* с именем `monad`.

Решение:

□

1.3 Светофорные дашборды

Разобравшись с деталями интеграции бизнес-юнитов корпорации, SRE-команда *Монокля* принялась за создание единой 24/7 дежурной смены и дашбордов для мониторинга всех сервисов корпорации. Чтобы дежурный мог быстро оценить состояние систем, главный дашборд в Grafana должен быть оформлен как множество *светофорных плиток* – каждая плитка Stat отображает состояние одного сервиса:

- Зеленая плитка OK означает, что все SLO сервиса выполняются.
- Желтая плитка WARN означает, что какой-то SLO сервиса находится в состоянии предупреждения.
- Красная плитка CRIT означает, что какой-то SLO сервиса не выполнен.
- Серая плитка ? означает, что какой-то из SLO не получилось посчитать (например, из-за No Data или ошибки запроса).

SRE-команда сразу же решила не заниматься накликиванием дашбордов вручную, а автоматизировать процесс создания дашбордов на основе шаблонов и API Grafana. Для этого они хотят написать службу, которая конвертирует описание SLO сервисов в красивый светофорный дашборд. Перед этим они разработали некое формальное исчисление для описания SLO сервисов корпорации, в него вошли следующие конструкции:

- $S_i :: service$ – сервис с номером i.
- $const :: scalar$ – можно объявить любую константу.
- $\{\text{UNK}, \text{CRIT}, \text{WARN}, \text{OK}\} :: state$ – монотонно возрастающее перечисление состояний многозначного SLO.
- $T_{99\%}(S_i) :: scalar?$ – время ответа (ms) в текущий момент у сервиса S_i для 99% пользователей.
- $R(S_i) :: scalar?$ – рейт запросов (rps) в текущий момент у сервиса S_i.
- $<:: scalar? \to scalar? \to state$ – оператор сравнения скалярных величин; если одна из величин неизвестна, возвращает UNK; если условие верно, возвращает OK; если условие неверно, возвращает CRIT.
- $\vee :: state \to state \to state$ – оператор объединения условий, возвращающий наилучшее из состояний; если одно из состояний UNK, то возвращает UNK.
- $\wedge :: state \to state \to state$ – оператор объединения условий, возвращающий наихудшее из состояний; если одно из состояний UNK, то возвращает UNK.

Так, корректным термом этого исчисления считается выражение типа $state$, например:

$$(T_{99\%}(S_1) < 200) \wedge (2000 < R(S_1)) \wedge (R(S_1) < 10000) :: state$$

$$(20 < 300) \vee \text{WARN} = \text{OK} :: \mathit{state}$$

Задание: Помогите SRE-команде *Монокля* реализовать транслятор их формализма в плитки Stat для дашборда в Grafana.

1. (0.25 б.) Запишите терм для следующего SLO: «99%-время ответа сервиса S_1 должно быть больше 200 мс для WARN и больше 300 мс для CRIT».

2. (0.5 б.) Отобразите в PromQL типы scalar и state и опишите, с помощью каких свойств Stat в Grafana необходимо раскрасить плитки в зависимости от значения state.

3. (0.75 б.) Покажите, как реализовать операторы $<, \vee, \wedge$ предложенного исчисления в PromQL (подсказка: используйте выражения вида `OR on() vector(0)`).

4. (1.5 б.) Предложите три SLO для параметров вашего ноутбука (docker-окружения), которые можно замерить с помощью `node-exporter`. В окружении Grafana с семинара создайте дашборд из трех светофорных плиток (необязательно Stat, но с раскраской) для них. Прикрепите три терма для ваших SLO, соответствующие им PromQL-выражения и скриншоты дашборда. Один из SLO должен содержать `WARN` уровень.

Решение:

□

1.4 Recording rules (*)

SRE-команда *Монокля* доказала свою компетентность в светофорных дашбордах, поразила своими скиллами все продуктовые команды и уже почти обрадовала топ-менеджмент. Но на демонстрации произошла неловкость.

На светофорной борде с интервалом $d = 5s$ было отображено более $n = 200$ светофорчиков, каждый из которых был реализован сложным PromQL-запросом с множеством операторов, рассмотренных выше. Prometheus не справлялся с этой нагрузкой, а Grafana подвисала при попытке отобразить дашборд. Топ-менеджмент был расстроен.

SRE-команда *Монокля* решила исправить ситуацию с помощью *recording rules*.

Задание: [1]

- (0.5 б.) Объясните, как использование *recording rules* поможет SRE-команде *Монокля* в данной ситуации.
- (1 б.) Напишите *rule group* для ваших метрик светофорных плиток из предыдущего задания в `prometheus.yml`. Проверьте корректность вашего файла с помощью `promtool` и продемонстрируйте результат. Оцените, насколько это субъективно ускорило загрузку вашего дашборда при малом интервале обновления.
- (0.5 б.) Какие ограничения/недостатки у использования *recording rules* вы можете назвать?

Решение:

□

[1] Задания и пункты, помеченные звездочкой, не являются обязательными. Эти баллы учитываются в общей сумме за курс, но являются дополнительными. Иными словами, вы можете сделать дополнительное задание вместо какого-то обязательного и получить тот же суммарный балл.

2 Нагрузочное тестирование

2.1 Закон Амдала

Предположим, что программа выполняет операцию широковещательной рассылки (broadcast). Эта операция вносит дополнительное время выполнения, зависящее от числа задействованных ядер n. Доступны две реализации broadcast:

- первая добавляет параллельные накладные расходы $\beta = 0.0001n$;
- вторая — $\beta = 0.0005 \cdot \ln(n)$.

Задание: (1.5 б.) Для какого числа ядер достигается наибольшее ускорение программы с долей последовательного кода$\alpha = 0.001$ в каждой из реализаций?

Решение:

□

2.2 Case study

В рамках этого задания вам предлагается выбрать систему, рассмотреть ее архитектуру, предложить оценку нагрузки и SLA, выявить *bottlenecks* (см. лекцию) и предложить метрики для мониторинга (см. задание 1.1).

Задание:

1. (1 б.) Опишите архитектуру и компоненты выбранной вами системы. Оцените внешнюю входную нагрузку и гарантии: throughput, latency, количество пользователей, объем хранимых данных. Прокомментируйте ваши оценки.

2. (1.25 б.) Выявите 3-5 потенциальных *bottlenecks* – проспекулируйте, в каких частях системы они могут возникнуть? Рассмотрите отдельно *database bottlenecks*.

3. (1.25 б.) Предложите метрики и их SLO для покрытия предложенных вами *bottlenecks*. Какие стратегии автоматического реагирования можно предложить в случае срабатывания алертов на них?

Решение:

□