

Санкт-Петербургский государственный университет

Кафедра системного программирования

Группа 21.Б11-мм

Разработка инструментария для обработки и визуализации сервисной информации

Илья Алексеевич Дудников

Отчёт по курсовой работе
в форме «Производственное задание»

Научный руководитель:
старший преподаватель кафедры системного программирования, Я. А. Кириленко

Консультант:
руководитель отдела серверных решений и СХД, ООО «КНС ГРУПП» А. А. Быков

Санкт-Петербург
2024

Оглавление

Введение	3
1. Постановка задачи	4
2. Обзор	5
2.1. Визуализация	5
2.2. Машинное обучение	9
3. Реализация	10
3.1. Анализируемые параметры	10
3.2. Получение и обработка данных	11
3.3. Визуализация данных	12
3.4. Предсказание отказа дисков	13
Заключение	16
Список литературы	17

Введение

В современном мире трудно переоценить значимость данных в бизнесе. Аккуратный анализ и структуризация этих данных позволяет компаниям принимать взвешенные решения и улучшать свои продукты, получая тем самым преимущество перед своими конкурентами.

Специалисты компании YADRO также сталкиваются с необходимостью извлечения полезной информации из большого количества данных. Так, инженеры технической поддержки проводят анализ различного рода сервисной информации о состоянии систем клиентов, обеспечивая их безотказную работу. Эта информация содержится, например, в *логах* и *снапах*. Логи представляют собой журнал событий, происходящих в системе (например, ошибки во время исполнения программ или действия пользователя), который в дальнейшем может быть использован для диагностирования возникающих проблем. В этой работе нас, однако, будут интересовать снапы. Снап — это архив с диагностической информацией о системе на какой-то момент времени. Однако информация в снапах слабо структурирована, в связи с чем их анализ становится затруднительным. Для этих целей разрабатывается инструмент *SnapShow*, позволяющий визуализировать такие данные, как характеристики накопителей, статистика по дискам (например, IOPS) и другие. Тем не менее инструмент позволяет получать лишь информацию по отдельным снапам, но не по всем системам (или подмножеству систем).

Поэтому поступил запрос на разработку инструмента, агрегирующего данные с различных систем. Он позволит производить более широкий анализ систем и выявлять типовые и проблемные компоненты и конфигурации оборудования. В рамках этой работы предполагается разработка этого инструмента.

1 Постановка задачи

Целью работы является разработка инструмент для обработки сервисной информации с целью выявления типовых и проблемных компонентов и конфигураций оборудования. Для её выполнения были поставлены следующие задачи:

1. выделить необходимые для анализа метрики;
2. спроектировать и разработать программу для выборки данных, а также базу данных для её хранения;
3. разработать инструмент для визуализации результатов;
4. внедрить методы машинного обучения для предсказания отказа дисков;
5. интегрировать полученные данные в имеющиеся средства аналитики.

2 Обзор

Поскольку задачи в ходе выполнения этой работы разделились на две относительно обособленные части, объединенные общей целью, этот раздел также будет разделен на два подраздела, посвященных визуализации и использованию машинного обучения соответственно.

2.1 Визуализация

В этом разделе мы рассмотрим существующие решения для визуализации данных. Мы обсудим их преимущества и недостатки, а также актуальность в контексте этой работы.

Grafana

Grafana [4] — это стек технологий для визуализации, мониторинга и анализа данных. В него входят такие продукты, как непосредственно Grafana (для визуализации данных), Grafana Loki [5] (для агрегации логов), Grafana Pyroscope [6] (для агрегации данных, полученных в результате профилирования) и другие. Предполагается, что наилучшим образом эти решения работают в связке, то есть как единое интегрированное решение.

Процесс получения данных и их преобразование для визуализации проиллюстрирован на рис. 1. Grafana поддерживает загрузку данных из множества различных источников, например, InfluxDB [7], Elasticsearch [3], MySQL [11] и другие. Кроме того, доступно большое количество плагинов, позволяющий загружать данные из других источников, для которых нет встроенной поддержки (на момент написания этой работы доступны более 150 плагинов). К каждому источнику данных также поставляется *редактор запросов*, позволяющий настроить, как и с какой частотой запрашивать данные из источника. Затем в разделе «Transform» можно указать преобразования для данных, если они необходимы. К таким преобразования можно отнести, например, переименование полей, агрегация результатов некоторых запросов, математические вычисления

и другие. Результаты преобразования также могут использоваться как входные данные для других преобразований.

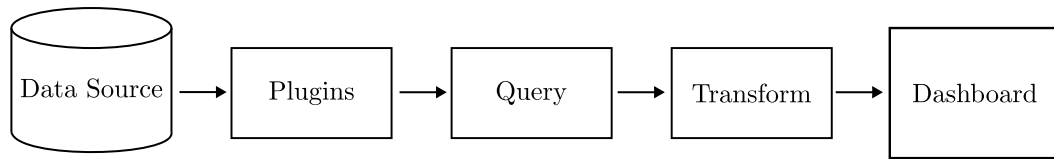


Рис. 1: Архитектура Grafana

После того, как данные получены и преобразованы, эти данные визуализируются. Grafana предоставляет широкий выбор способов визуализации, такие как круговые, столбчатые диаграммы, таблицы, гистограммы и многие другие. Пример панели, которую позволяет построить Grafana, представлен на рис. 2.

Кроме того, Grafana предоставляет возможность настраивать *оповещения* в случае, если происходят какие-то (заранее определенные инженером) события. Эта функциональность избавляет программиста от необходимости непрерывного наблюдения состояния системы.

Как можно заметить, основным предназначением Grafana является мониторинг системы. В рамках нашей же работы необходима визуализации редко изменяющихся данных. И хотя Grafana можно использовать и для таких целей, наличие избыточной и неиспользуемой функциональности может привести к использованию большего количества ресурсов, чем необходимо. Важно сказать, что в случае Grafana использование лишних ресурсов не критично, поскольку она менее требовательна, чем, к примеру, Kibana [8], поэтому Grafana остается предпочтительным вариантом в случае, если желаемая функциональность в разрабатываемом инструменте потребует использование более сложных решений.

Наконец, Grafana оптимизирована для работы с *временными рядами* (time series). Нас же интересует лишь последняя доступная информация о системах, а не их изменение во времени.

Kibana

Kibana [8] — это продукт из стека ELK (Elasticsearch [3], Logstash [10], Kibana), предназначенный для визуализации данных. Аналогично рас-



Рис. 2: Пример панели, сделанной с помощью Grafana

смотренной ранее Grafana, в Kibana доступны различные способы загрузки данных. Получение данных из источников осуществляется с помощью раздела «Discover», а для фильтрации данных используется *Kibana Query Language* [9]. Одним из ключевых отличий Kibana от Grafana является отсутствие встроенной функциональности для преобразования данных. Эти преобразования можно осуществлять в других продуктах из стека (Elasticsearch и Logstash), что, однако затрудняет использование Kibana для визуализации в изоляции от других продуктов. Кроме того, она лучше всего приспособлена для работы с временными рядами, хотя не ограничивается только ими. Пример панели, построенной с помощью Kibana, представлен на рис. 3.

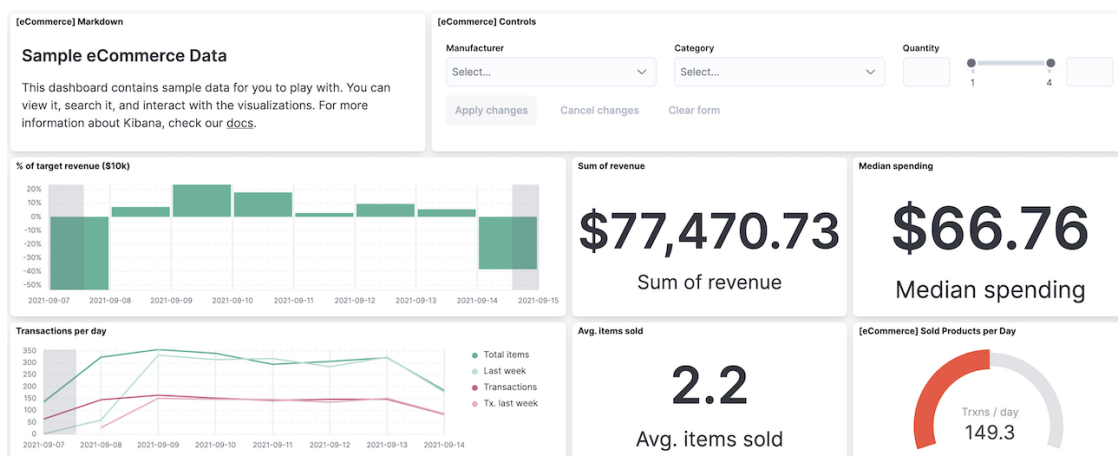


Рис. 3: Панель в Kibana

Также ELK предоставляет встроенные функции для использования машинного обучения. Например, функция *выявления аномалий* моделирует нормальное поведение системы, основываясь на данных за прошедшее время, что позволяет ей определять моменты времени, когда возникают непредвиденные и/или вызывающие ошибки события. Возможно также настроить оповещения о нахождении аномалий. К сожалению, эти функции платные, что на текущий момент не позволяет воспользоваться их преимуществами. К тому же, для использования машинного обучения в Kibana необходим Elasticsearch.

В общем случае, вероятно, нет проблемы в использовании всего стека целиком, или хотя бы интегрированного решения в виде Elasticsearch и Kibana. Для наших текущих требований, однако, накладные расходы на ресурсы, необходимые для интегрирования Elasticsearch, не оправданы. Впрочем, в рамках этой работы разработка инструмента выполняется с предположением, что может возникнуть потребность в использовании и таких инструментов, как Kibana или описанная ранее Grafana, даже если на текущем этапе такой необходимости нет.

Plotly и Plotly Dash

Plotly [13] — это набор библиотек для построения интерактивных графиков. Самой популярной и распространенной из них является библиотека для языка Python, но доступны также и решения для других

языков, таких как, например, R, JavaScript, F# и прочих.

Поскольку Plotly выполняет лишь построение графиков, программист не ограничен в том, что использовать в качестве источника данных и как эти данные трансформировать, что является безусловным преимуществом в сравнении с описанными выше решениями. Ограничение возникает лишь из-за количества и разнообразия доступных видов графиков, где Plotly не уступает Kibana и Grafana.

Plotly также предоставляет фреймворк *Plotly Dash* [14], абстрагирующий написание HTML разметки и позволяющий построить панель визуализации целиком на Python. Это, в том числе, позволяет делать панели более интерактивными, чем возможно в других решениях. Поскольку разметка страницы задается динамически, появляется возможность отслеживать различные события и реагировать на них изменениям графика или даже нескольких графиков. К примеру, такими событиями могут быть нажатия кнопки мыши или наведение курсора на элементы графиков. Ещё одной возможностью является кросс-фильтрация, то есть фильтрация одновременно нескольких графиков, если они к примеру, отражают связанные данные (или одинаковые данные, но с разных перспектив).

2.2 Машинное обучение

Optuna

Optuna [12] — это легковесный фреймворк с открытым исходным кодом, предназначенный для эффективного подбора гиперпараметров моделей. Подбор гиперпараметров модели — это одна из наиболее критических частей машинного обучения. Однако она трудоемка и неэффективна, если подбор выполняется вручную. Optuna позволяет автоматизировать этот процесс, осуществляя поиск оптимальных параметров в заданном пользователем пространстве решений (search space). Кроме того, доступно несколько различных алгоритмов подбора, начиная с полного перебора до TPE и CMA-ES.

3 Реализация

3.1 Анализируемые параметры

Анализировались следующие параметры систем.

- MTM (machine-type & model).
- Дисковые метрики:
 - FRU part number;
 - physical capacity;
 - transport protocol.
- Метрики RAID:
 - raid level;
 - strip size;
 - physical capacity;
 - physical free capacity;
 - allocated capacity;
 - effective used capacity.

- Microcode version.

Кратко опишем значения этих полей. «MTM» (Machine type & model) — это номер вида XXXX-YYY, где XXXX — это номер «machine type», а YYY — номер «model». У машин с одинаковыми MT, и тем более одинаковыми MTM, схожие спецификации. «FRU part number» (Field replacement unit part number) — это номер, позволяющий при необходимости найти заменяемой части подходящую замену (в нашем случае заменяемая часть — это диск). «physical capacity» соответствует физическому размеру диска. Значение «transport protocol» сообщает транспортный протокол диска. Это поле имеет одно из двух значений: «sas» или «nvme».

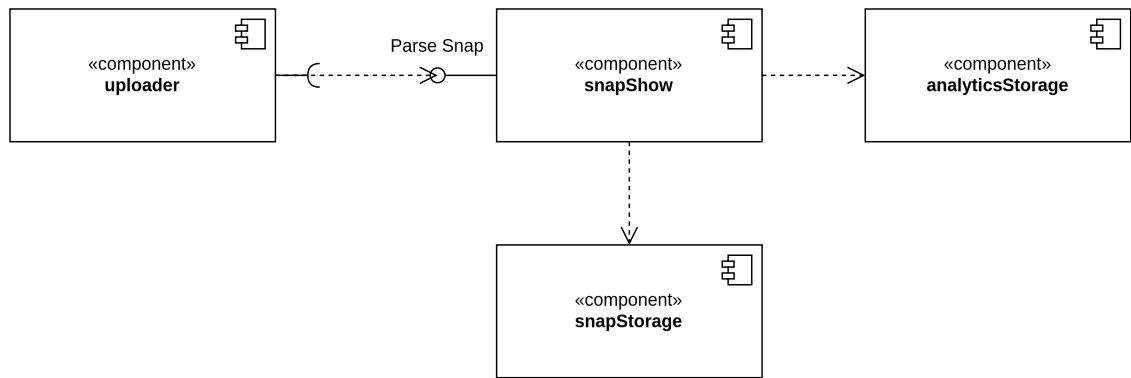


Рис. 4: Получение и обработка данных

Следующие данные относятся к конфигурации RAID массивов. Поля «raid level» и «strip size» показывают уровень RAID массива и размер страйпов (если используется striping). «physical capacity», «physical free capacity», «allocated capacity» и «effective used capacity» хранят различную информацию о емкости массива. «physical capacity» и «physical free capacity» — это общий объем и объем незанятого физического пространства, доступного массиву, соответственно, в то время как «allocated capacity» и «effective used capacity» соответствуют объемам логического пространства (общего и занятого, соответственно).

Наконец, «microcode version» соответствует версии микрокода отдельного вычислительного узла машины.

3.2 Получение и обработка данных

Диагностическая информация хранится в MongoDB в неструктурированном виде, откуда её берут все сервисы, работающие с ними (в первую очередь SnapShow). При загрузке очередного снапа данные из него парсятся и загружаются в MongoDB. Затем каждую ночь SnapShow забирает из базы данных снапы, поступившие за день, разбирает в нем описанные выше метрики и загружает их уже в структурированном виде в PostgreSQL. На рис. 4 представлена диаграмма, описывающая этот процесс. Также на рис. 5 представлена диаграмма PostgreSQL.

Такой подход был выбран в качестве компромисса между актуальностью данных и скоростью работы сервиса. В нашем случае нет необ-

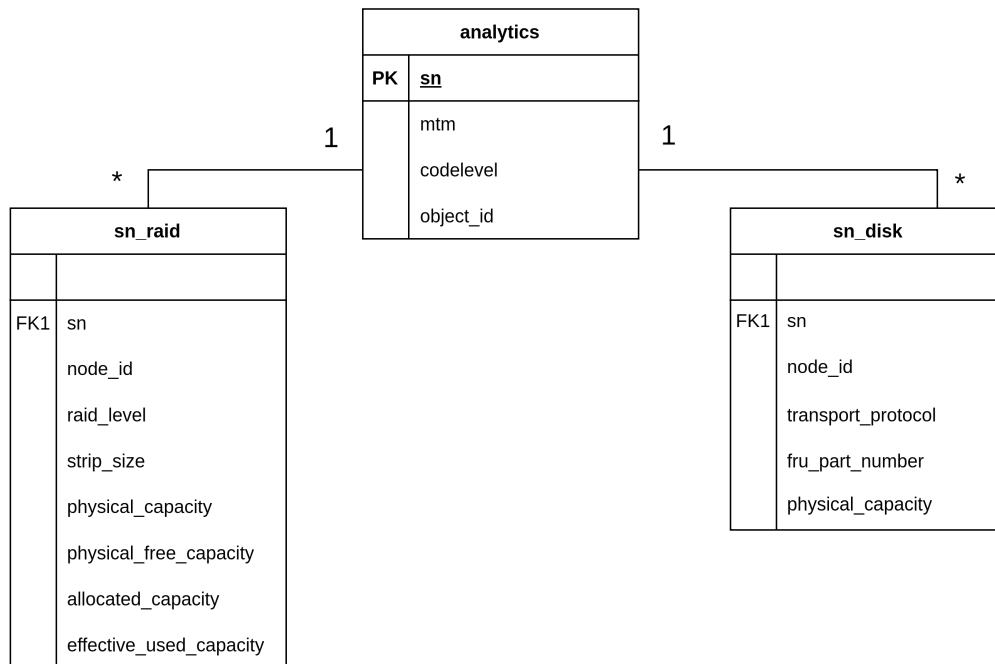


Рис. 5: Схема базы данных для хранения визуализируемых данных

ходимости поддерживать данные в абсолютно актуальном состоянии и их обновление раз в день вполне приемлемо. В то же время обновление данных непосредственно при загрузке лишь увеличит нагрузку на сервис и скажется на другой функциональности сервиса, где актуальность данных действительно важна.

3.3 Визуализация данных

Для визуализации данных использовалась библиотека Plotly. Графики строились на основе данных из PostgreSQL, где они хранятся в практически готовом для построения формате. Строились следующие диаграммы.

- Круговая и столбчатые диаграммы для категориальных метрик:
 - MTM;
 - Microcode version;
 - FRU Part number;
 - raid level;

- и др. за исключением метрик, соответствующих емкостям дисков или RAID массивов.
- Гистограммы для численных метрик:
 - physical capacity;
 - physical free capacity;
 - allocated capacity;
 - effective used capacity.

Также для каждого из описанных графиков добавлена возможность визуализации какого-то подмножества систем на основе МТ и МТМ. Иными словами, возможно просматривать графики для машин с конкретным МТ или МТМ.

3.4 Предсказание отказа дисков

Диски имеют один из трех возможных статусов:

- «online», означающий, что диск полностью исправен;
- «offline», означающий, что диск недоступен;
- «degraded», означающий, что диск недопустен через несколько портов.

Наше предположение заключалось в том, что данные о конфигурации дисков и статистика их работы позволит обоснованно предсказывать, что диск сломается. Это позволит инженерам проверить возможные неполадки до того, как диск откажет. Для достижения этой цели использовалось машинное обучение, а именно: решалась задача классификации с тремя классами, где целевой признак — это статус диска, а нецелевые — параметры конфигурации и статистики дисков. Статистики работы дисков, о которых идет речь, описаны в таблице 1.

Для подбора параметров использовался фреймворк Optuna. Для оценки качества работы модели использовалась метрика `f1_macro`. Её

Название статистики	Описание
ro	Количество операций чтения
wo	Количество операций записи
rb	Количество считанных блоков данных
wb	Количество записанных блоков данных
re	Суммарное время ответа на запрос чтения
we	Суммарное время ответа на запрос записи
rq	Суммарное время ответа на запрос чтения с учетом времени ожидания в очереди
wq	Суммарное время ответа на запрос записи с учетом времени ожидания в очереди
pro	Максимальное время ответа на запрос чтения с учетом времени ожидания в очереди
pwo	Максимальное время ответа на запрос записи с учетом ожидания в очереди
pre	Максимальное время ответа на запрос чтения
pwe	Максимальное время ответа на запрос записи

Таблица 1: Описание статистик дисков

Модель	Время обучения, с	f1_macro
HistGradientBoostingClassifier	0.399	0.691
XGBClassifier	0.408	0.672
CatBoostClassifier	11.626	0.666

Таблица 2: Результаты выбора модели

выбор обусловлен, во-первых, тем, что, нам важен как precision, так и recall (поэтому f1), и, во-вторых, естественной несбалансированностью классов (естественной, потому что в нормальных условиях исправных дисков должно быть значительно больше, чем всех остальных), что привело к выбору среднего «macro». Для предотвращения переобучения при подборе параметров и оценке качества использовать 5-fold кросс-валидация.

Модель выбиралась среди моделей из scikit-learn [15], xgboost [2] и catboost [1]. В таблице 2 представлены три модели, которые показали лучший результат.

Как можно видеть, HistGradientBoostingClassifier показала себя лучше всего, поэтому на момент написания этой работы используется именно она.

Заключение

В рамках этой производственной практики были успешно выполнены следующие задачи.

- Выделены метрики для анализа.
- Разработана программа для сбора данных и база данных для их хранения.
- Реализован инструмент для визуализации полученных данных.
- Внедрено предсказание отказа дисков на основе методов машинного обучения.
- Полученные результаты интегрированы в имеющиеся средства аналитики.

Что касается дальнейшего развития инструмента, можно выделить следующие направления.

- Применение схожих методов машинного обучения для предсказания статуса RAID массивов, а также отдельных вычислительных узлов.
- Использование статистики о работе дисков для выявления аномалий в работе дисков.
- Анализ выживаемости дисков для расширения предсказания отказов временным интервалом до отказа.
- Использование статистики для диагностики других возможных проблем с дисками.

Список литературы

- [1] Catboost. — URL: <https://catboost.ai/>.
- [2] Chen Tianqi, Guestrin Carlos. [XGBoost: A Scalable Tree Boosting System](#) // Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. — KDD '16. — New York, NY, USA : ACM, 2016. — P. 785–794. — URL: <http://doi.acm.org/10.1145/2939672.2939785>.
- [3] Elasticsearch. — URL: <https://www.elastic.co/elasticsearch>.
- [4] Grafana. — URL: <https://grafana.com/>.
- [5] Grafana Loki. — URL: <https://grafana.com/oss/loki/>.
- [6] Grafana Pyroscope. — URL: <https://grafana.com/oss/pyroscope/>.
- [7] InfluxDB. — URL: <https://www.influxdata.com/>.
- [8] Kibana. — URL: <https://www.elastic.co/kibana>.
- [9] Kibana Query Language. — URL: <https://www.elastic.co/guide/en/kibana/current/kuery-query.html>.
- [10] Logstash. — URL: <https://www.elastic.co/logstash>.
- [11] MySQL. — URL: <https://www.mysql.com/>.
- [12] [Optuna: A Next-generation Hyperparameter Optimization Framework](#) / Takuya Akiba, Shotaro Sano, Toshihiko Yanase et al. // Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. — KDD '19. — New York, NY, USA : Association for Computing Machinery, 2019. — P. 2623–2631. — URL: <https://doi.org/10.1145/3292500.3330701>.
- [13] Plotly. — URL: <https://plotly.com/>.
- [14] Plotly Dash. — URL: <https://dash.plotly.com/>.

- [15] Scikit-learn: Machine Learning in Python / F. Pedregosa, G. Varoquaux, A. Gramfort et al. // Journal of Machine Learning Research. — 2011. — Vol. 12. — P. 2825–2830.