Санкт-Петербургский Государственный Университет
Математико-механический факультет
Кафедра системного программирования

Чугаев Анатолий Александрович

Визуализация данных тестирования storage-систем

Курсовая работа

Научный руководитель:
ст. преп. Немешев М.Х.

Санкт-Петербург
2018

Оглавление

1.Введение

В современную эпоху объем хранимой и передаваемой информации растет в невероятно огромной прогрессии. В связи с этим, возникает необходимость надежного хранения данных и быстрого доступа к ним. Для того чтобы обеспечить storage-системы этими качествами в Dell-EMC существует специальный отдел, состоящий из “performance” инженеров, который занимается тестированием этих систем, для дальнейшего их улучшения. Однако, для более качественного анализа результатов автоматического тестирования инженерам нужно их визуальное представление (в виде графиков, таблиц и т.д.).

Для этого был создан легко расширяемый framework “PerfStat”. С помощью данного framework’а можно составить полноценный отчет о полученных данных тестирования.

Используя данный framework, разработчик получает расширяемый и легко повторно используемый и настраиваемый инструмент, что позволяет экономить время на разработку, а также может быть встроен в автоматизированный процесс тестирования.

В настоящее время инженеры используют самописные скрипты или крупные системы аналитики (Tableau, QlikView, Microsoft Power BI). Скрипты сложно переиспользовать другим инженерам, а аналитические системы слишком избыточные и громоздкие по функциональности, совершенного не адаптированы под семантику данных и не имеют возможности встраивания в процесс тестирования.

2.Цели и задачи

Задачей проекта являлось создание framework’а для анализа и визуализации данных тестирование storage-систем. Моей основной задачей являлось создание архитектуры представления данных внутри разрабатываемого framework’а, а также разработка пользовательского интерфейса.(Рис. 1)

Для этого было необходимо решить следующие подзадачи:

- Изучение предметной области:
  - изучение систем хранения данных
  - изучение нагрузочного тестирования
- разработать составные типы данных для внутреннего представления данных тестирования для различных форматов
- разработать общую объектную модель для дальнейшей визуализации данных для следующих структурных элементов отчета:
  - график
  - таблица
  - текст
- разработать конфигурационный файл для взаимодействия пользователя и фреймворка
  - разработать формат файла

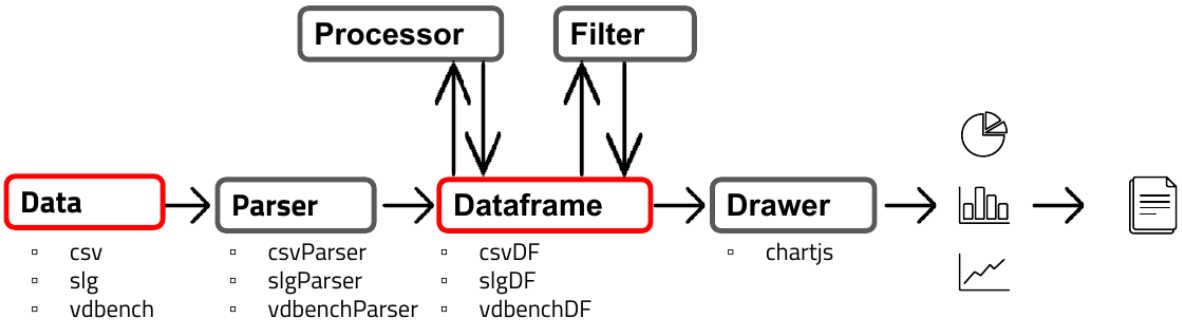

Рис. 1 Архитектура всего фреймворка, мои задачи - реализация бока Data и Dataframe

3.Обзор предметной области

Storage-система - хранилище, которое позволяет пользователям постоянно хранить и извлекать цифровые данные.[3]

4.Реализация

4.1 Разработка общей объектной модели

Первым делом были разработаны классы общей объектной модели. Было принято решение о создании базового визуального элемента от которого бы наследовались другие. Это решение позволило сделать архитектуру более гибкой, для того чтобы была возможность добавить какие-либо новые элементы визуализации в будущем.

Далее были выделены главные сущности отчета: текст, таблица, график. Для каждой из них был создан класс: текст - Text, таблица - Table, а для графика было принято решение выделить три класса из которых будет строится сама картинка. Класс Axis для осей координат, Series для описания одной линии, Chart для описания общей диаграммы. (Рис. 2)

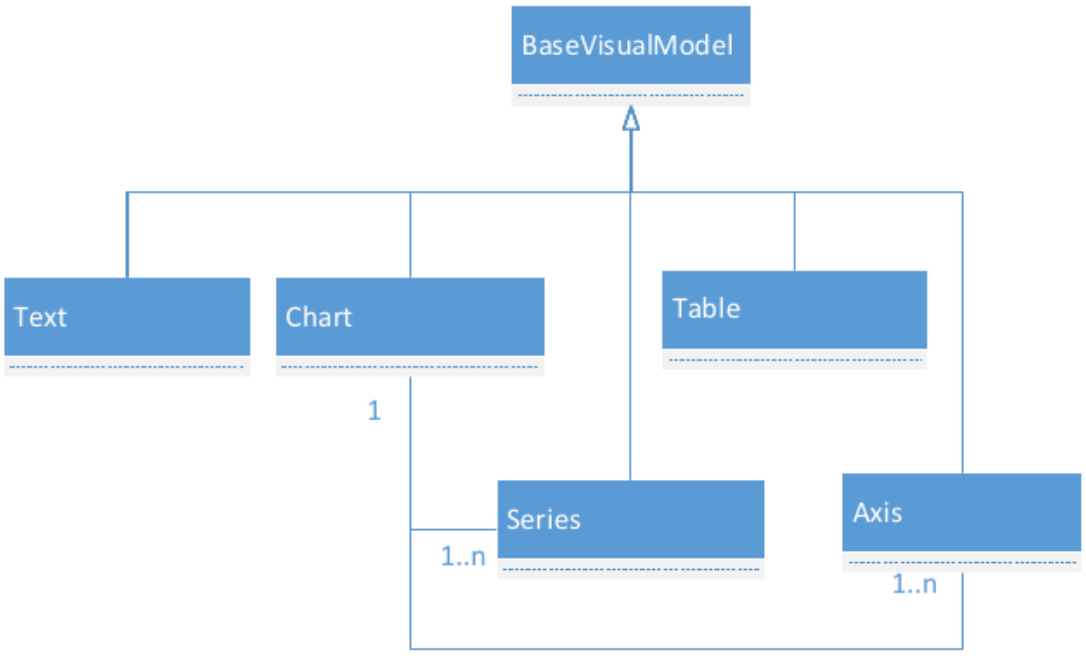

Рис. 2 Итоговая диаграмма классов объектной модели

4.2 Разработка внутреннего представления данных

Также как и случае с разработкой общей объектной моделью, нужно было обеспечить расширяемость системы в будущем для каких-либо других форматов данных тестирования.

Первым делом был создан базовый класс BaseDataFrame в основе которого лежит библиотека pandas[5], от которого наследуются классы для других форматов данных: CSVDataFrame - для CSV, SLGDataFrame - для SLG, VDBENCHDataFrame - для vd bench.(Рис. 3)

При разработке архитектуры для общей объектной модели и модели внутреннего представления данных использовались методологии из следующих ресурсов[1],[2].

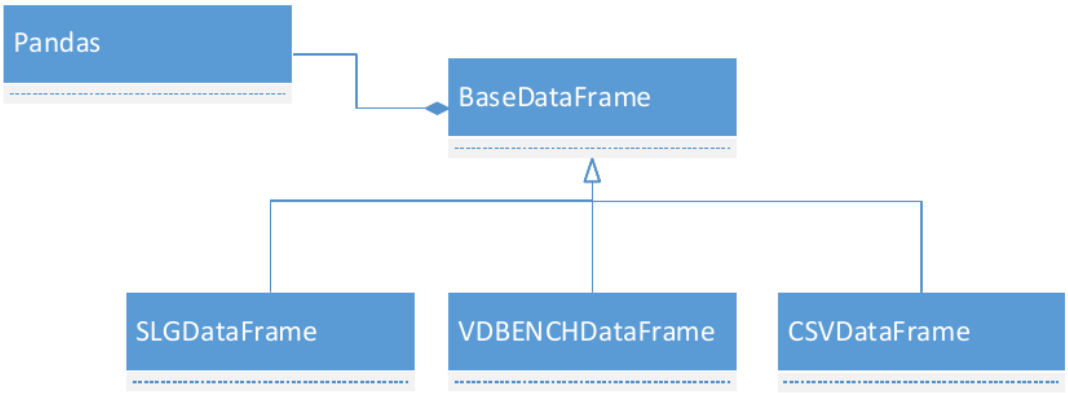

Рис. 3 Итоговая диаграмма классов представления данных

4.3 Разработка конфигурационного файла

В качестве формата конфигурационного файла был выбран формат JSON[4]. Причин для этого было несколько: легко “парсить”, простой синтаксис для написания разработчиком, хорошо подходит для описания сущностей объектной модели.

Конфигурационный файл состоит из нескольких блоков: "data" - здесь указывается какой файл с данными выбрать (есть поддержка регулярных выражений для задания имени файла), а также какие фильтры и агрегирующие функции применить к данным; "series_templates" - шаблон для описания формата одного графика (line, bar, pie, polarArea и т.д.) по одному набору данных и его атрибутов (цвет, подпись, ширина и т.д.); "axes_templates" - шаблоны для описания системы отсчета;
"table" - описание таблиц, "text" - описание текстовых комментариев, "tasks" - главный блок который состоит из списка заданий, каждое задание представляет собой элемент из блока "data" и шаблоны которые применяются для визуализации этих данных.

В секции "tasks" для каждого задания пользователь может выбрать два мода: "one" - много графиков на одном canvas, "many" - на каждый график свой canvas.

5.Заключение

В ходе работы достигнуты следующие результаты:

- разработаны составные типы данных для внутреннего представления данных тестирования следующих форматов:
  - csv
  - slg
  - vdbench
- разработана общая объектная модель для дальнейшей визуализации данных, состоящая из следующих сущностей:
  - Series
  - Axis
  - Chart
  - Text
  - Chart
- разработан конфигурационный файл для взаимодействия пользователя и фреймворка
  - форматом файла выбран JSON

Список литературы

[1] Len Bass, Paul Clements, Rick Kazman - Software Architecture in Practice (3rd Edition) - https://www.amazon.com/Software-Architecture-Practice-3rd-Engineering/dp/0321815734

[2] Roger S. Pressman, Bruce Maxim - Software Engineering: A Practitioner's Approach (8th Edition) - https://www.amazon.com/Software-Engineering-Practitioners-Roger-Pressman/dp/0078022126

[3] Information Storage and Management: Storing, Managing, and Protecting Digital Information in Classic, Virtualized, and Cloud Environments (2nd Edition) - EMC - 2012

[4]Introducing JSON - https://www.json.org/

[5]Documentation Pandas - https://pandas.pydata.org/pandas-docs/stable/