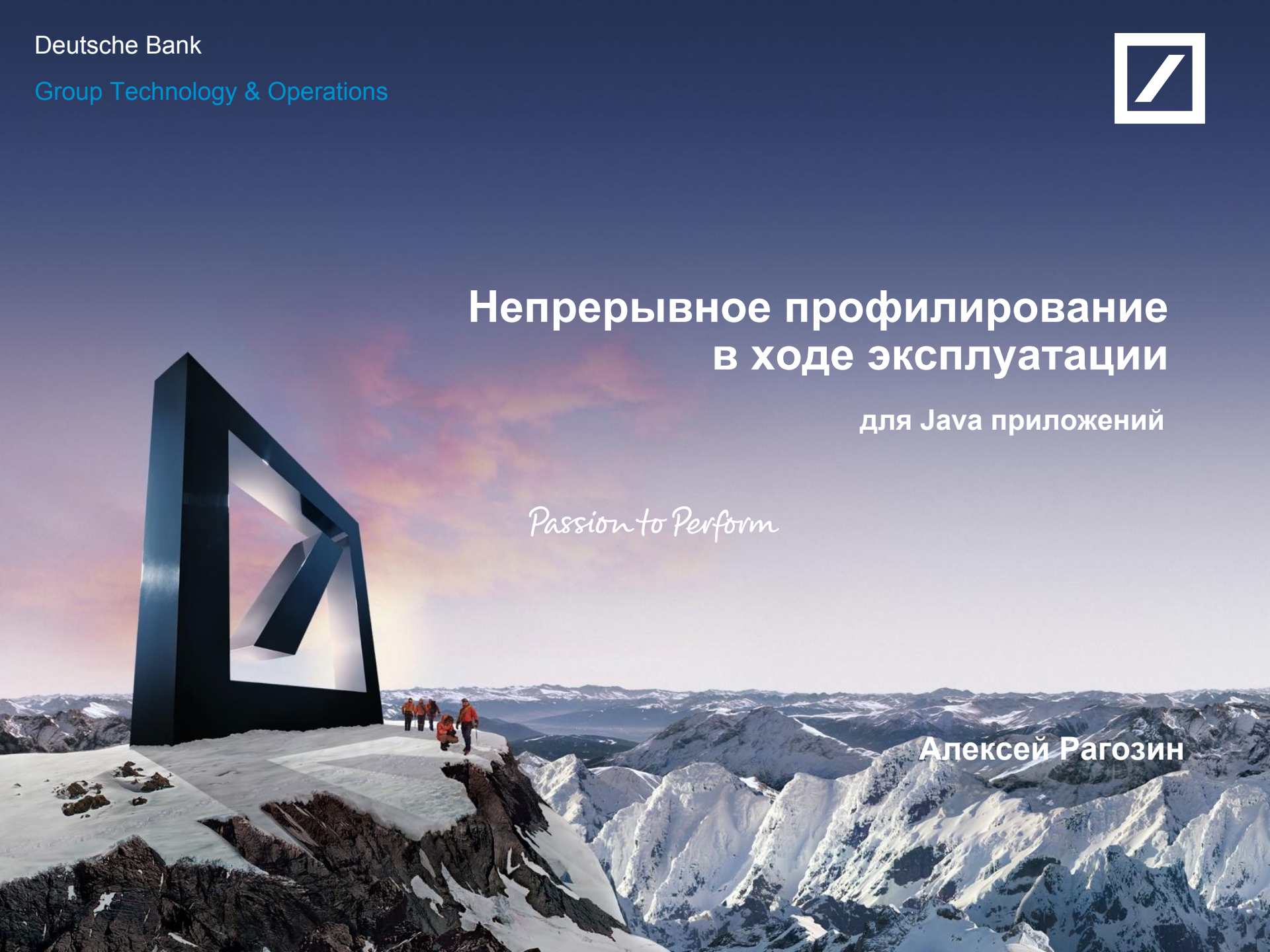
Deutsche Bank

Group Technology & Operations



Непрерывное профилирование в ходе эксплуатации

для Java приложений

Passion to Perform

Алексей Рагозин

Профилирование

Java даёт нам богатейший набор инструментов

- Широкий выбор профайлеров
- Диагностические интерфейсы JVM
  - ✓ JVMTI, инструментация, JMX и т.д.

Но есть небольшая проблема…

Профилирование

Как воспроизвести проблему в условиях эксперимента?

Нам нужно

- Реальное железо
- Реальные данные
- Реальная нагрузка

Плюс, очень чёткое понимание того что же именно происходит в ходе эксплуатации!

Terra incognita

Что же всё-таки наше приложение делает на самом деле?

Можно спросить

- нам не ответят / соврут

Можно замерить

…

Метрики

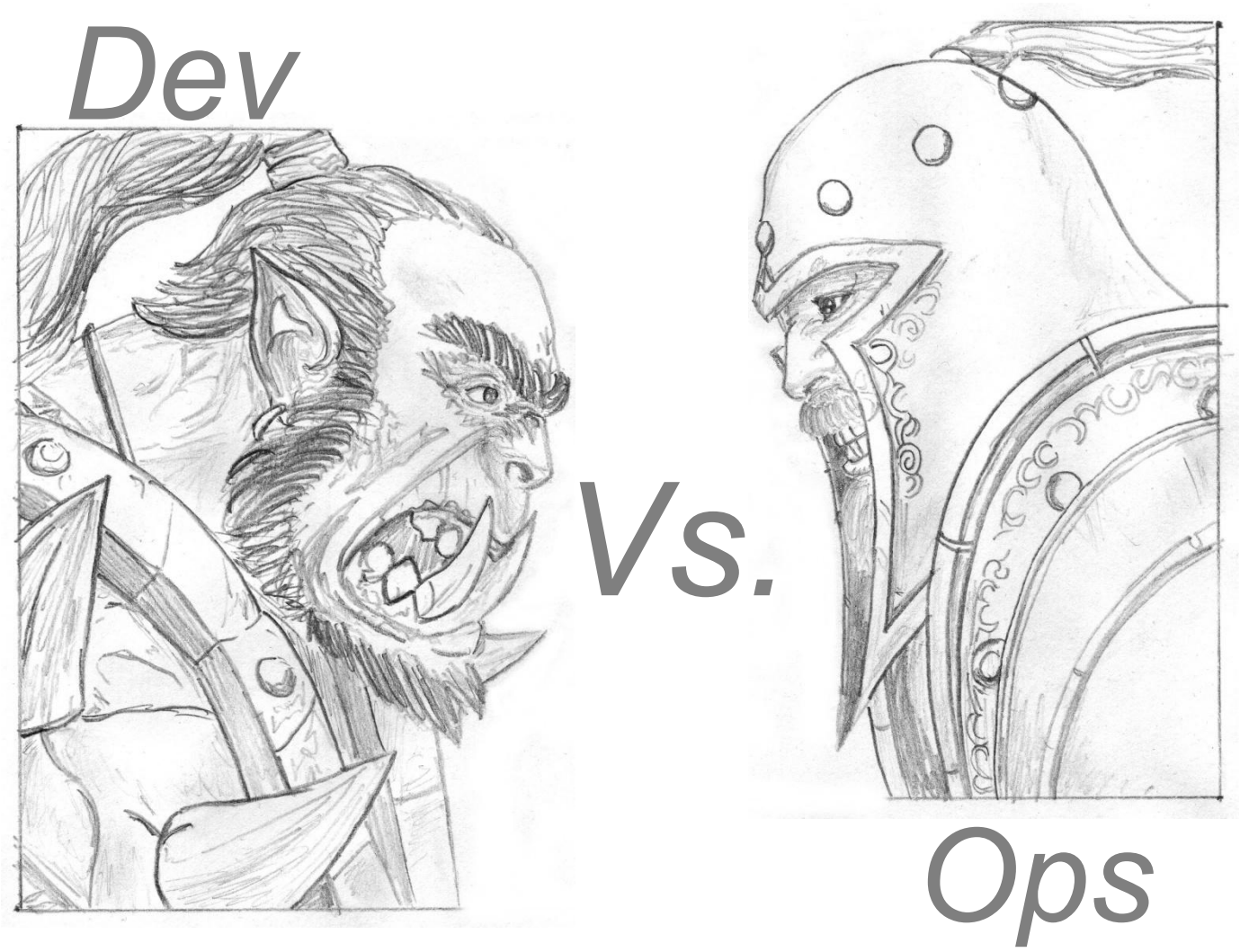
Метрики

Замеряем / Агрегируем / Публикуем

- Perf4j - http://perf4j.codehaus.org/
- Simon - https://github.com/virgo47/javasimon
- Metrics - https://dropwizard.github.io/metrics/
- MoSKito - http://www.moskito.org/
- Servo - https://github.com/Netflix/servo

and counting ...

Метрики



Метрики

Мониторинг

Профилирование

Телеметрия

Результаты прямых измерений

Не агрегированные

Пригодные для статистической обработки

Включая

- замеры специфические для приложения
- общее состояние системы (ЦПУ, сеть и т.п.)

Данные обрабатываются ретроспективно

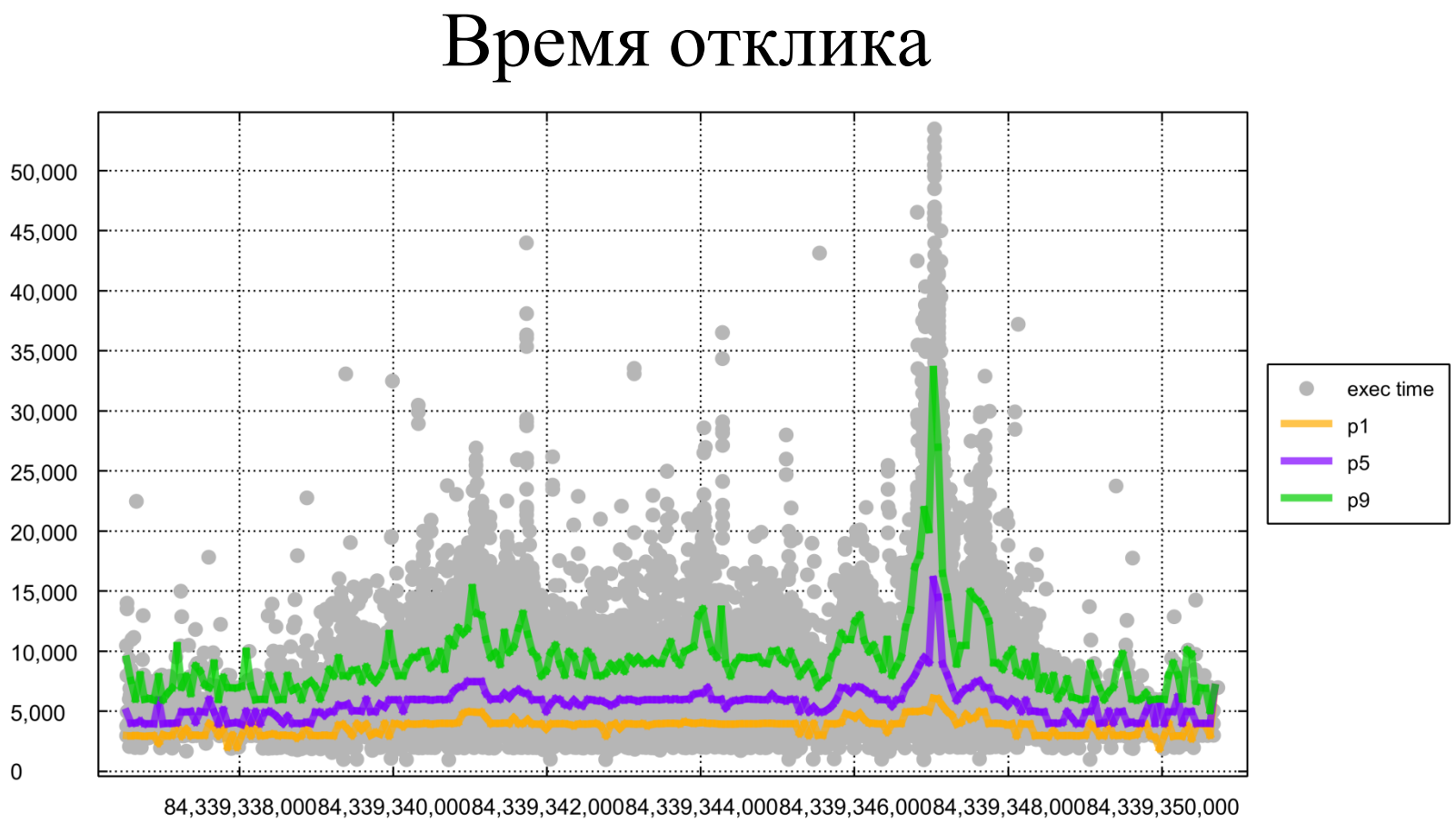
Телеметрия

Big data?

- Текстовая строчка лога ~ 150 байт
- Бинарно упакованная строчка ~ 30 байт
  - ✓ Сжимается в 2-3 раза
- 10 миллионов событий в день - *это много!*
  - ✓ 100-200 MiB в день

Работа с данными

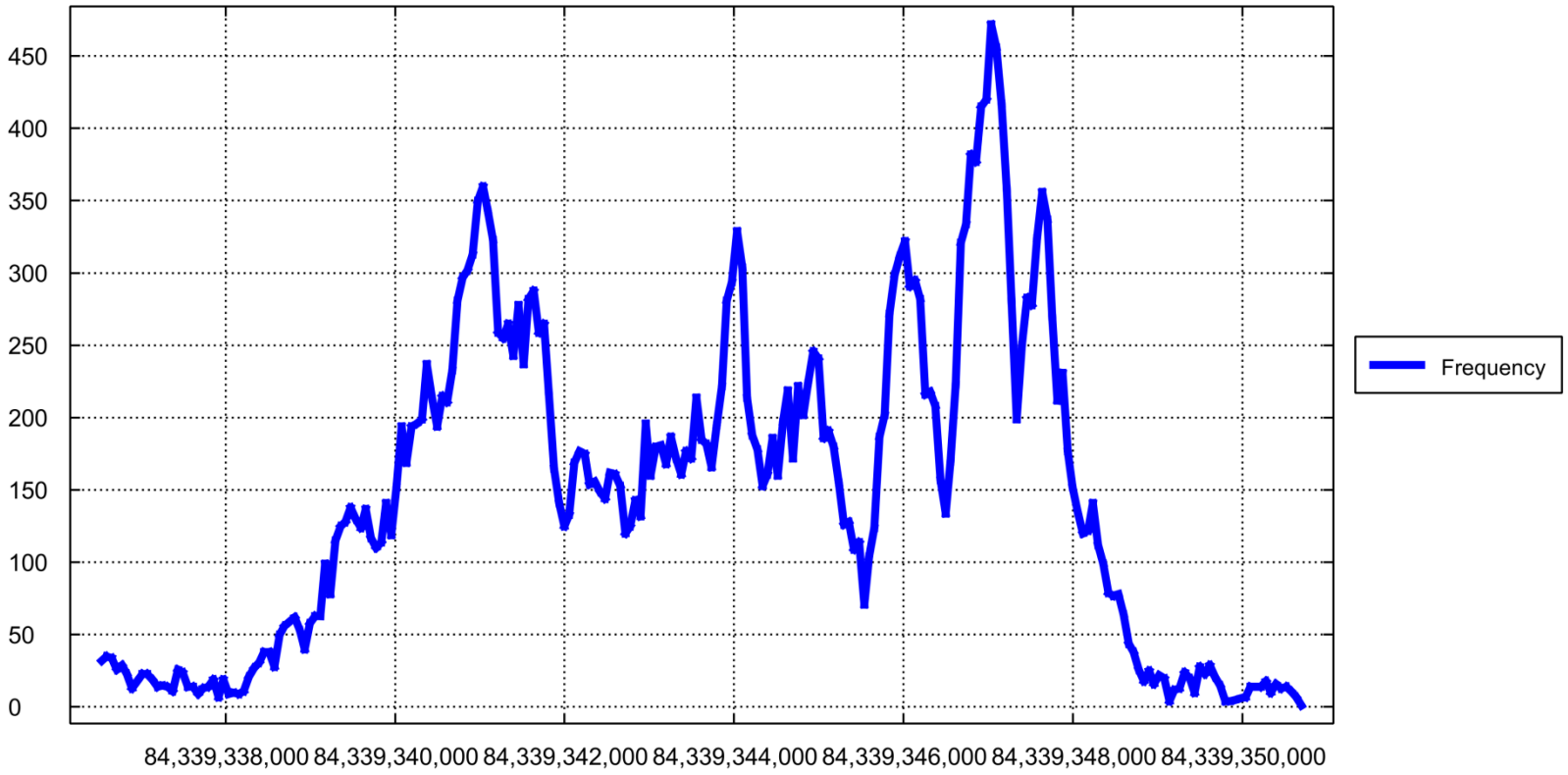
KPI Pivoting



Работа с данными

KPI Pivoting

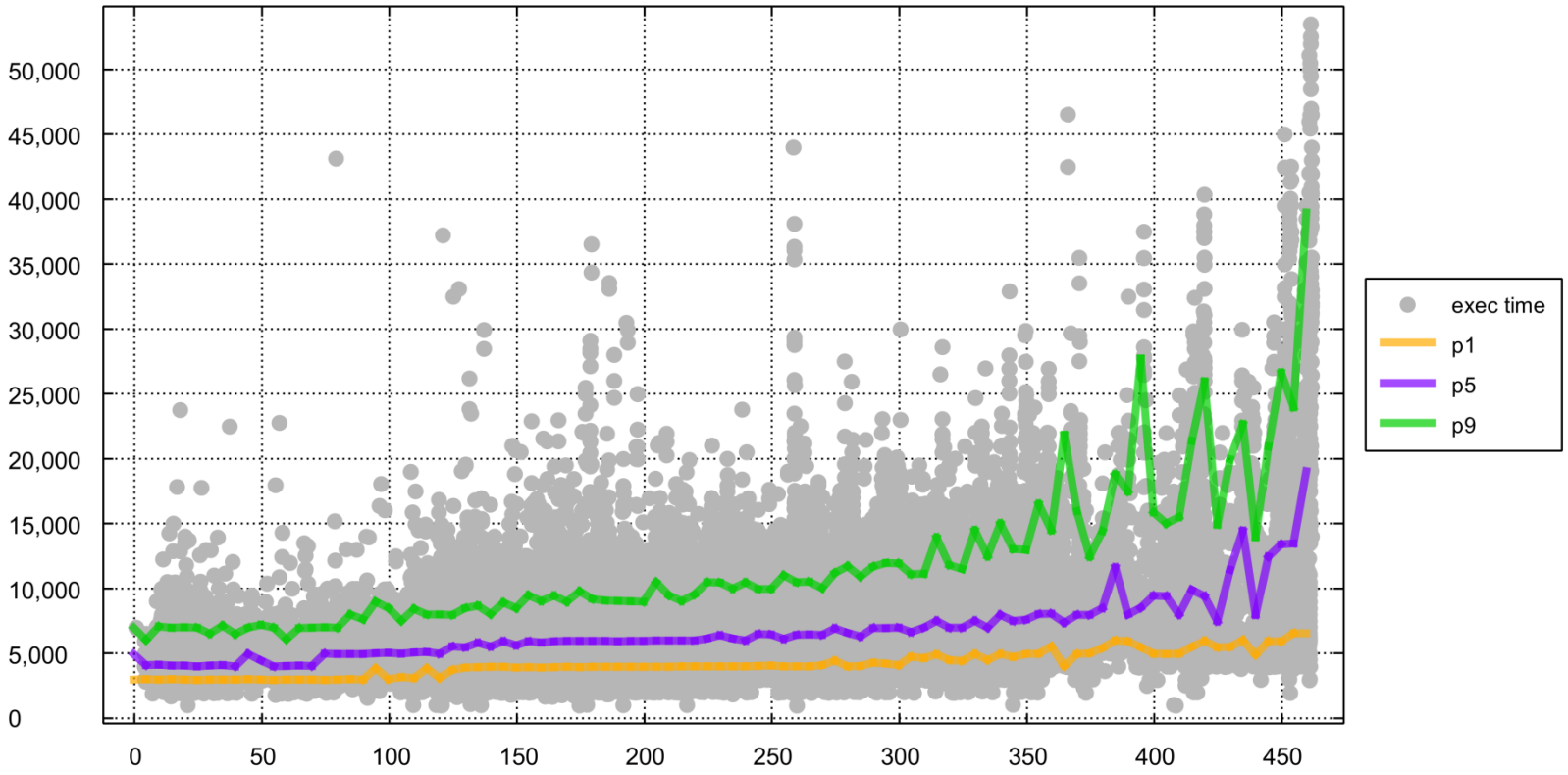
Частота запросов



Работа с данными

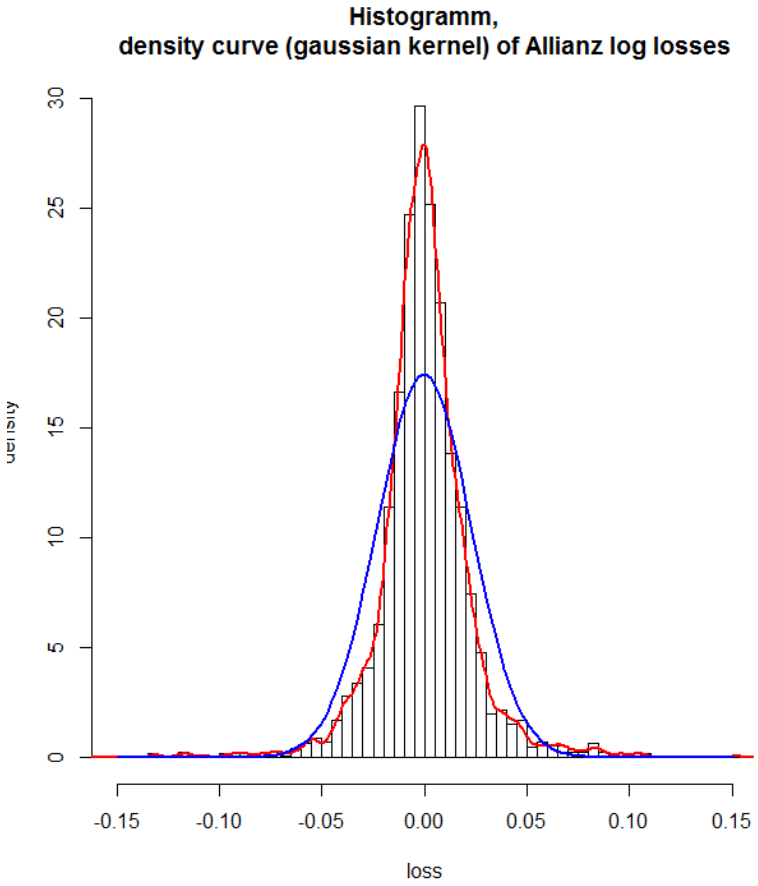
KPI Pivoting

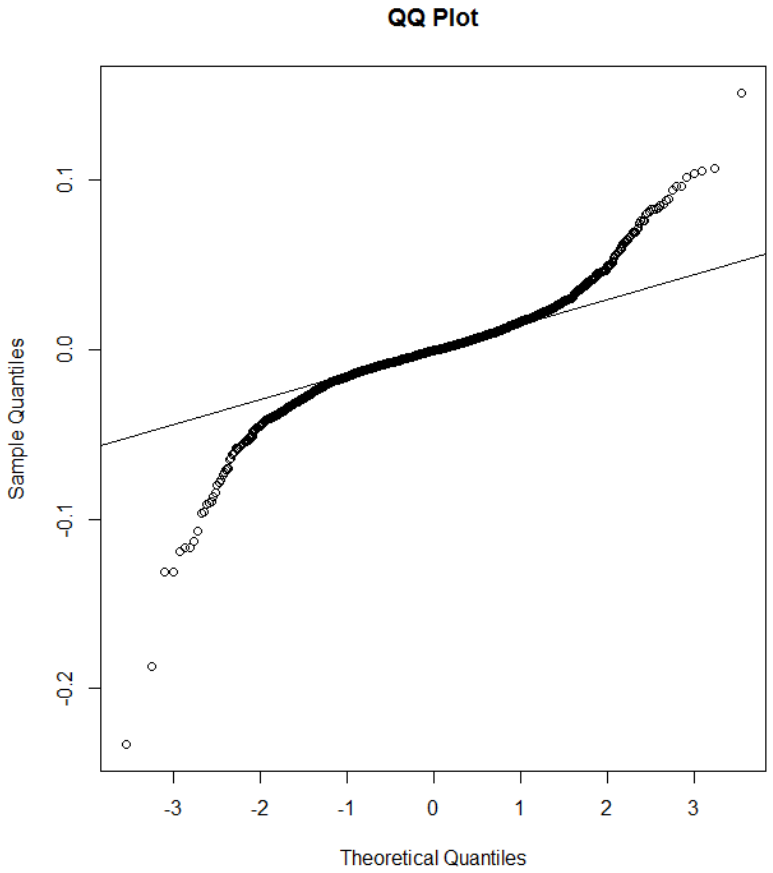
Время отклика от частоты



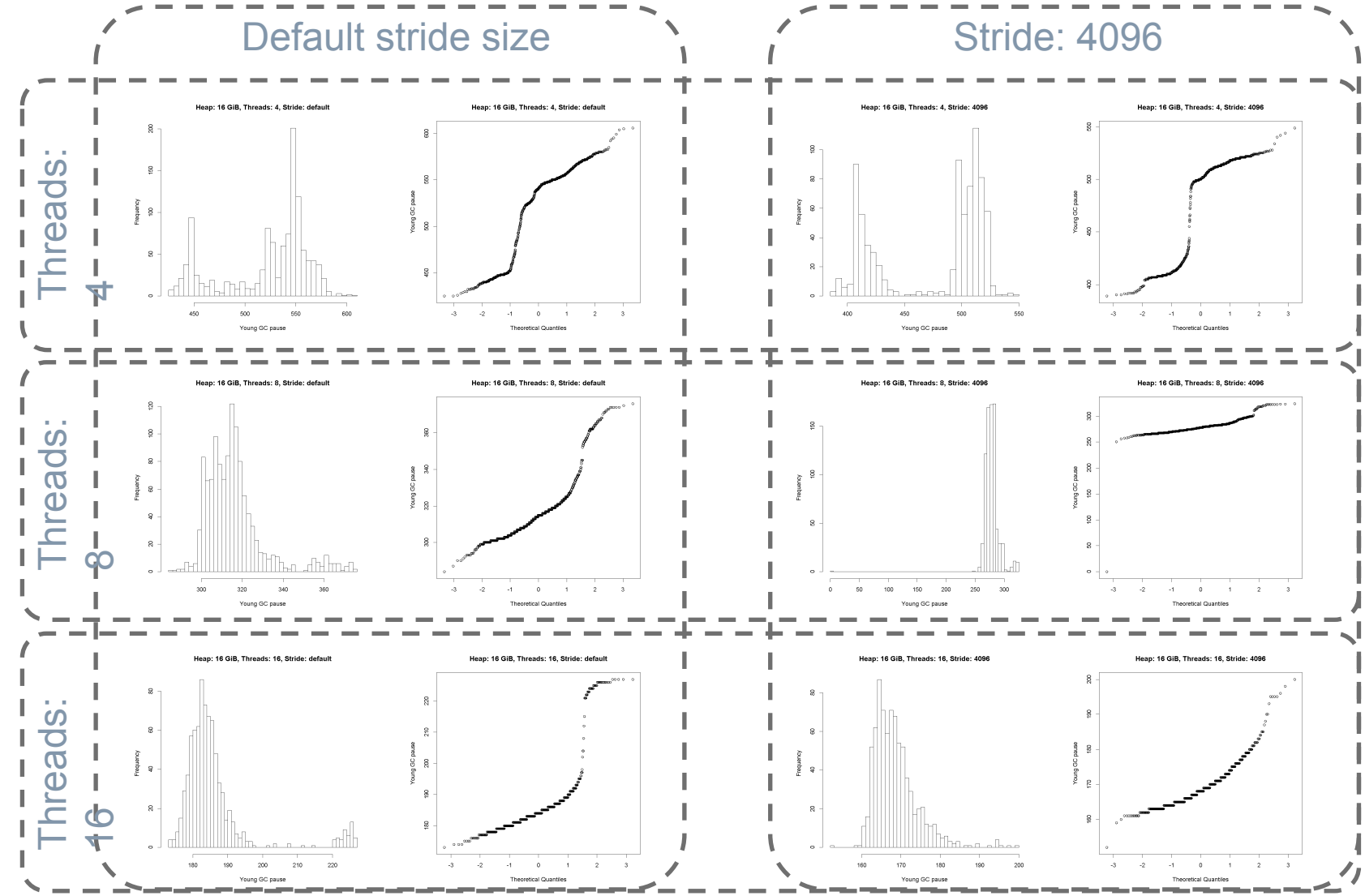
Работа с данными

Quantitive Quintile

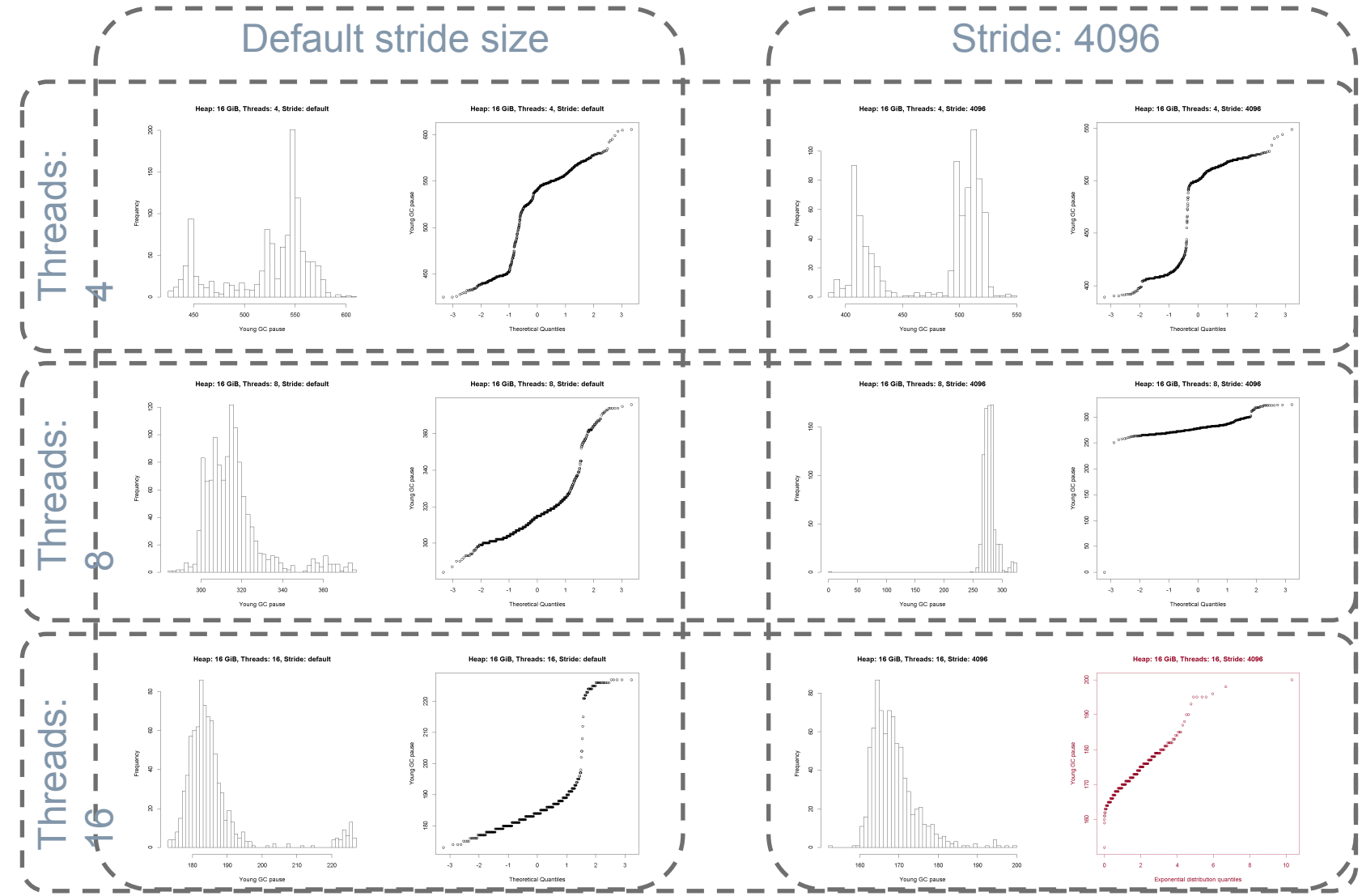




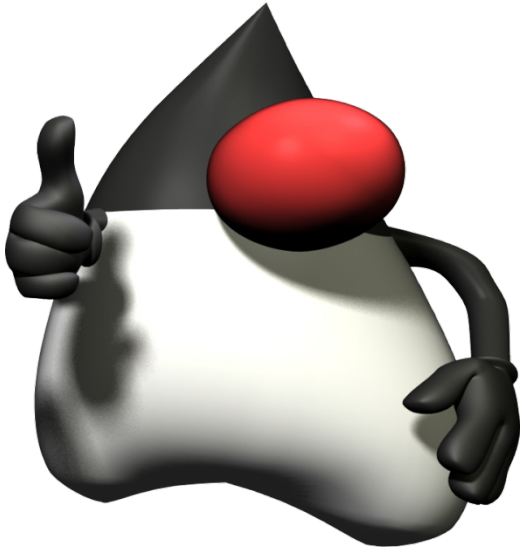
Quantitive Quintile



Quantitive Quintile



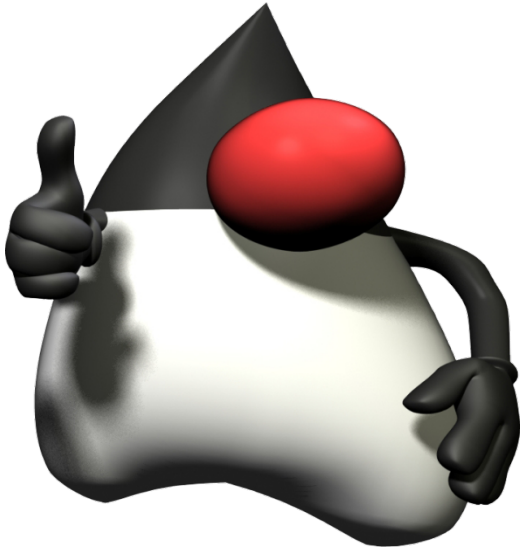
Работа с данными



Работа с данными

•меньше 1000 строк
•пивот таблицы
- Интерактивная работа с данными

•QQ диаграммы
•статистичекие “навороты”



•любые объёмы данных
•сложные расчёты
- предагрегация
- автоматизация

Профилирование

Профилирование

Инструментация (“логи”)

- Правильный выбор точек замеров
- Крупная гранулярность

Сэмплирование

- Не требует конфигурации
- Проблемы выявляют себя сами

Профилирование
Сэмплирование

JVM использует safepoint
для получения дампа потоков

- Нагрузка на JVM
- Особенности расстанови safepoint`ов

Сэмплер стек трейсов не использующий safepoint`ы
https://github.com/RichardWarburton/honest-profiler

Профилирование
Сэмплирование

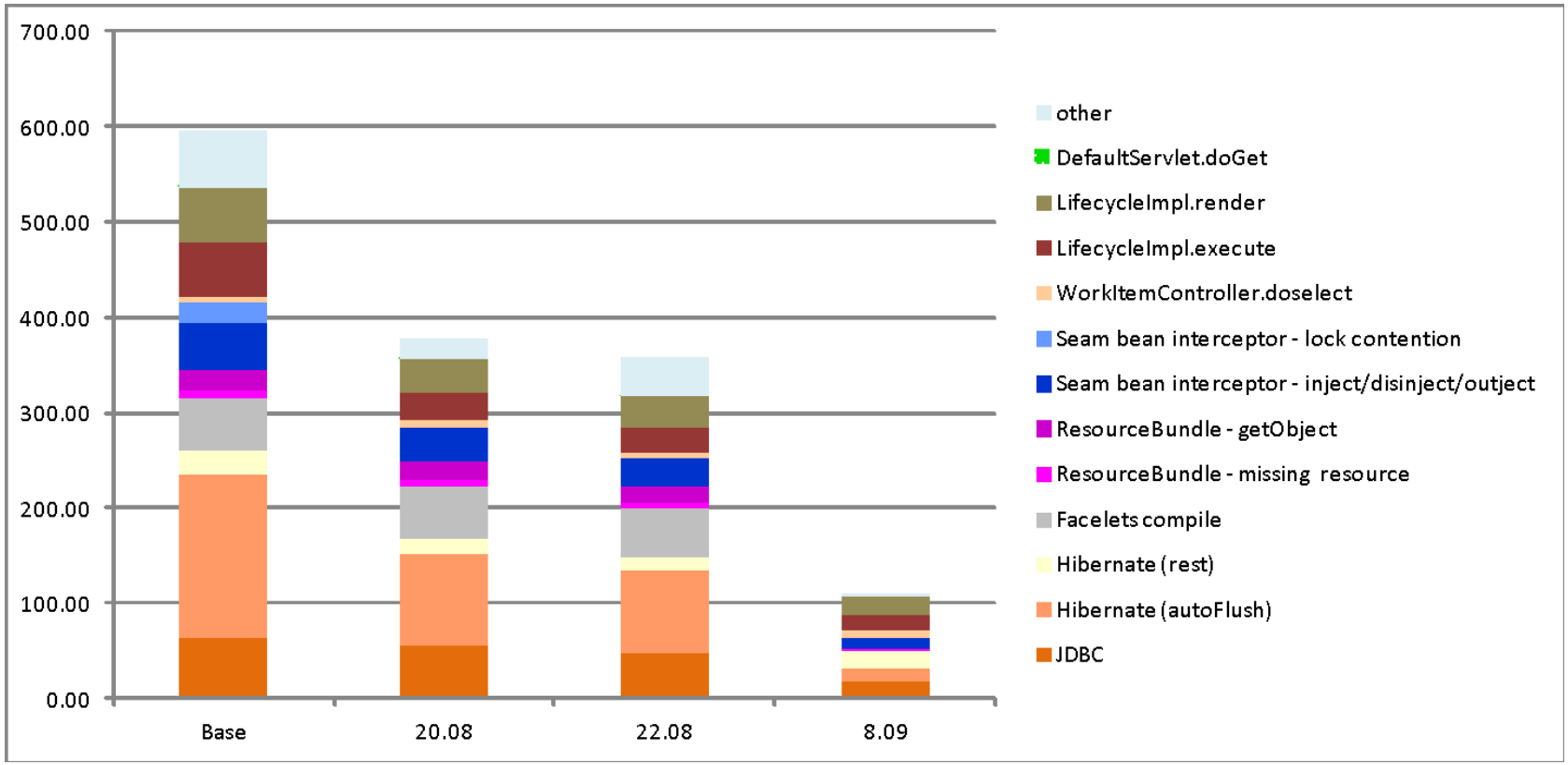
1 стектрейс – не значит ничего

Набор стектрейс – даёт на вероятности

Условные вероятности

– ключевой момент анализа

Профилирование
Сэмплирование



Профилирование
Сэмплирование

Гистограмма стек фреймов

```
...
316209  28%  1065911 com.sun.faces.facelets.impl.DefaultFacelet.include(DefaultFacelet.java:379)
316209  28%  1065908 com.sun.faces.facelets.impl.DefaultFacelet.include(DefaultFacelet.java:320)
316129  28%  4096200 javax.faces.view.facelets.DelegatingMetaTagHandler.apply(DelegatingMetaTagHandler.java
316129  28%  316129  com.sun.faces.facelets.tag.jsf.core.ViewHandler.apply(ViewHandler.java:188)
316129  28%  3150391 javax.faces.view.facelets.CompositeFaceletHandler.apply(CompositeFaceletHandler.java:9
313448  28%  3945843 org.richfaces.view.facelets.html.BehaviorsAddingComponentHandlerWrapper.applyNextHandl
313448  28%  3945843 com.sun.faces.facelets.tag.jsf.ComponentTagHandlerDelegateImpl.apply(ComponentTagHandl
313448  28%  3869592 javax.faces.view.facelets.DelegatingMetaTagHandler.applyNextHandler(DelegatingMetaTagH
309959  28%  309959  org.hibernate.ejb.QueryImpl.getResultList(QueryImpl.java:264)
299384  27%  299384  org.hibernate.internal.SessionImpl.autoFlushIfRequired(SessionImpl.java:1204)
296253  26%  803071  com.sun.faces.facelets.tag.ui.IncludeHandler.apply(IncludeHandler.java:120)
291880  26%  369667  org.jboss.seam.util.Reflections.invoke(Reflections.java:22)
289383  26%  289383  org.hibernate.internal.SessionImpl.list(SessionImpl.java:1261)
288139  26%  353384  org.jboss.seam.intercept.RootInvocationContext.proceed(RootInvocationContext.java:32)
287339  26%  352555  org.jboss.seam.transaction.RollbackInterceptor.aroundInvoke(RollbackInterceptor.java:2
287331  26%  352581  org.jboss.seam.intercept.SeamInvocationContext.proceed(SeamInvocationContext.java:56)
281061  25%  343119  org.jboss.seam.core.BijectionInterceptor.aroundInvoke(BijectionInterceptor.java:79)
264960  24%  484726  com.sun.faces.facelets.tag.ui.CompositionHandler.apply(CompositionHandler.java:166)
258967  23%  270125  com.sun.faces.facelets.impl.DefaultFaceletContext.includeDefinition(DefaultFaceletCont
...
```

Профилирование
Сэмплирование

В ходе эксплуатации

- Один дамп в секунду
- 3600 в час
- 28800 за рабочий день
- 144000 в неделю

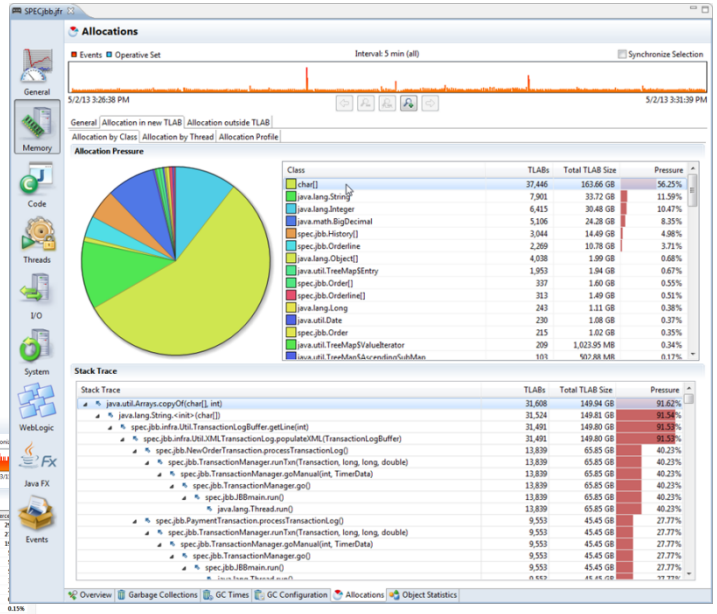
- Достаточно для анализа
  - 30 байт на стрейс X количество потоков в дампе
  - 30 x 24 x 7 x 3600 x 50 ≈ 900 MiB в неделю

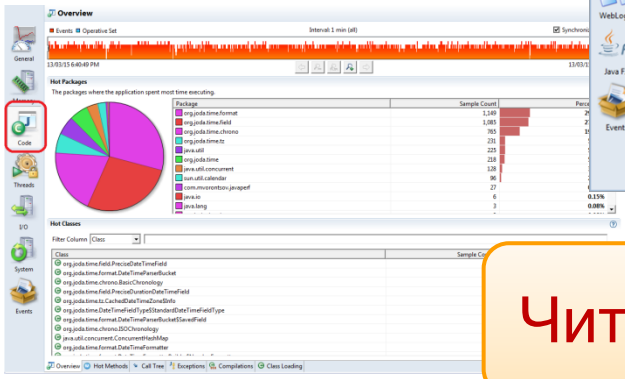
https://github.com/aragozin/jvm-tools

Профилирование

Mission Control / Flight recorder

- Компакт бинарный формат
- Легковесный
- Различные метрики JVM
- Сэмплирование потоков
- Контекстное сэмплирование



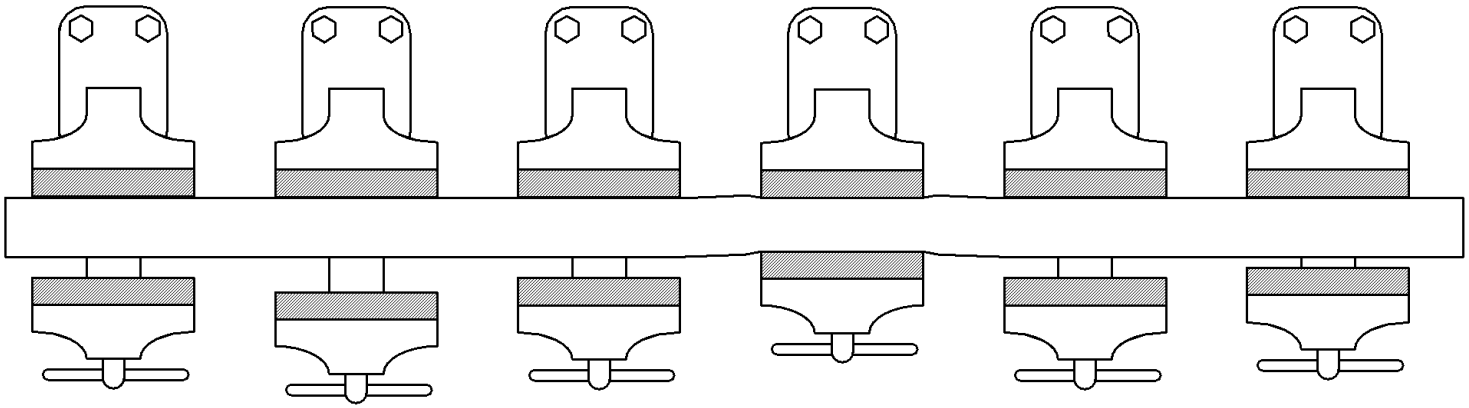


Читайте условия лицензии!

В заключение

- Вам нужны метрики с продакшена
  хотя бы, чтобы возпроизвести проблему.
- Чуть больше информации и этого может быть
  достаточно чтобы идентифицировать её причину.

В заключение



Вашем приложении сотни узких мест,
но лишь одно из них является проблемой, которую надо решать именно сейчас.

СПАСИБО

Алексей Рагозин

alexey.ragozin@gmail.com