



От Статистики до Статистики

Докладчик:

Юхно Олег

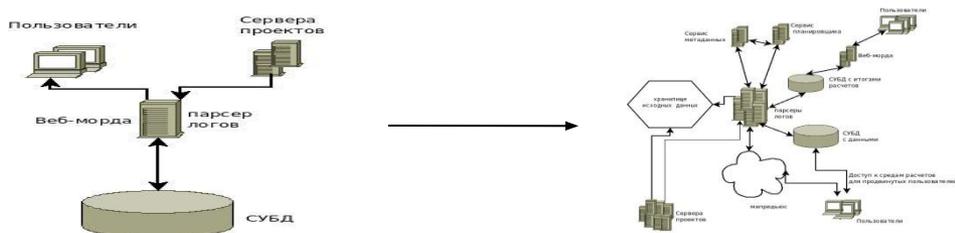
руководитель группы администрирования

email: olegu@yandex-team.ru



О чем я хочу рассказать?

Развитии системы статистики



Проблемах и их решении



Зачем нужна система расчета статистики



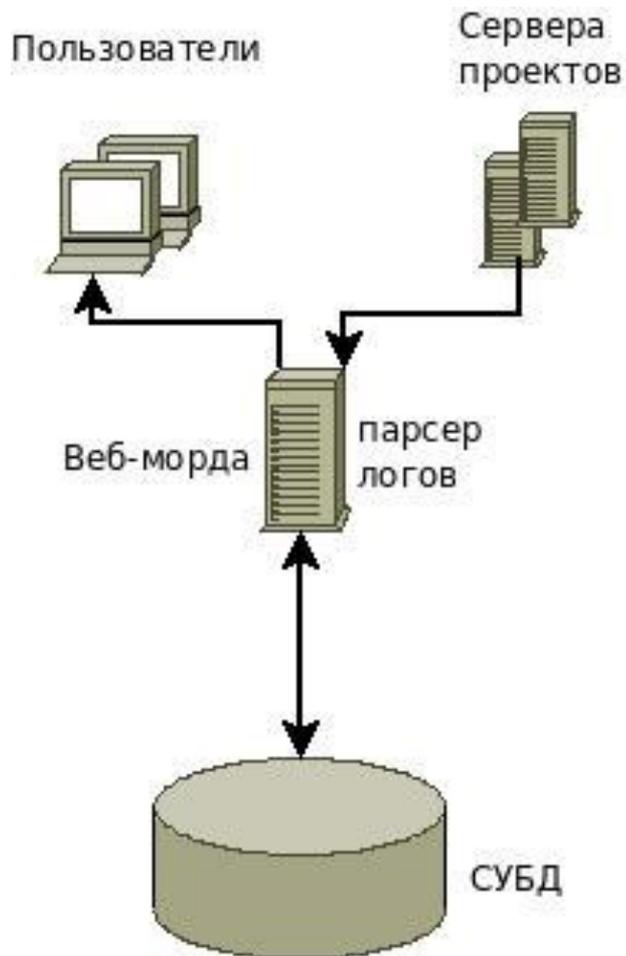
Я

Почему нужна система расчета статистики

- Много проектов
- Много данных
- Потребовалось считать статистику регулярно и единообразно



Сначала данных и проектов было мало



< 10Gb в сутки

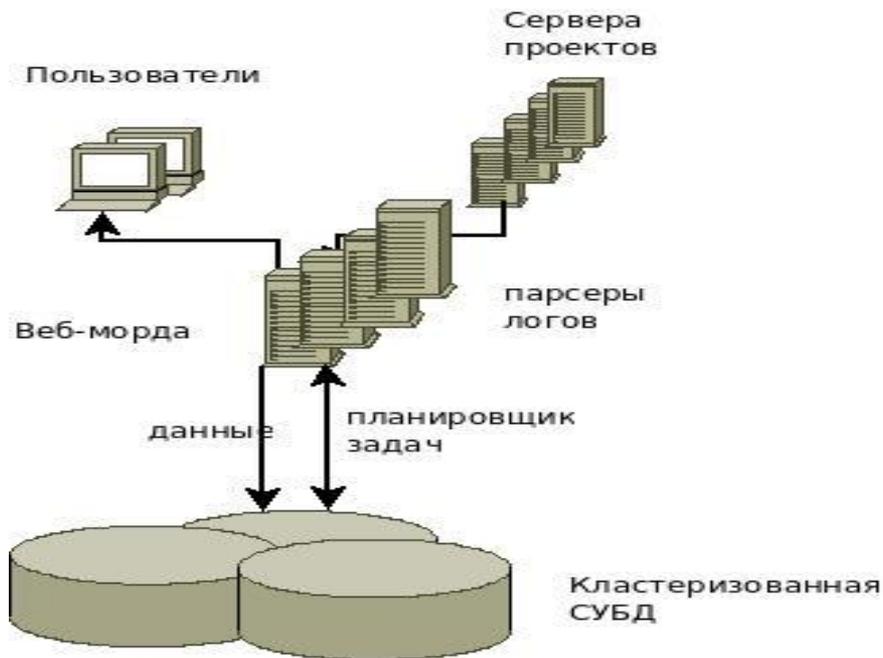
Я

Исходная архитектура

- в СУБД складываем только необходимые фрагменты строк из лога
- СУБД резервируем
- считаем возможности экстенсивного роста неограниченными
- **Проблема:** конструкция не масштабируется.



Данных и проектов еще было мало



< 100Gb в сутки

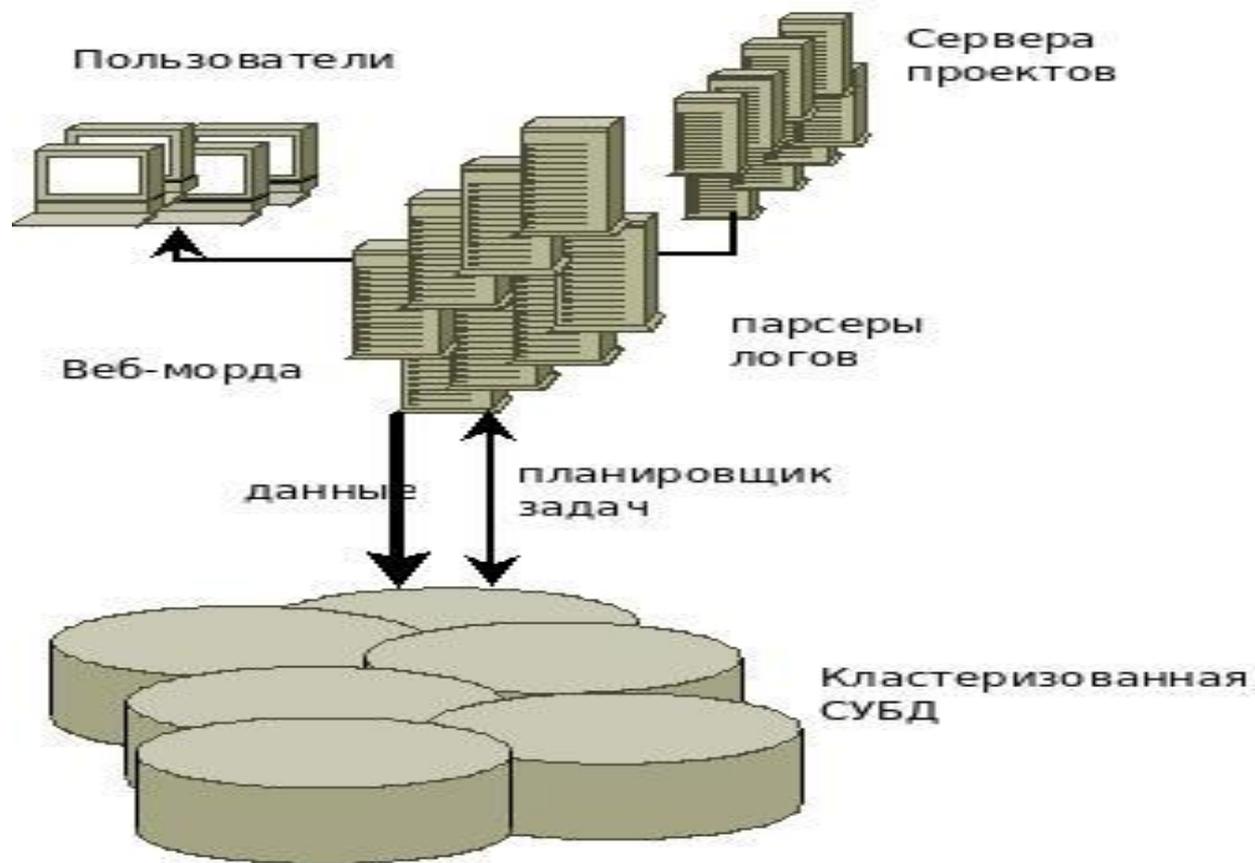


Первая попытка масштабирования

- кластеризация считающих машин ("парсеров")
- кластеризация СУБД
- SAN с использованием iscsi
- Проблема: все пропало, сеть не может обеспечить нужную производительность :(



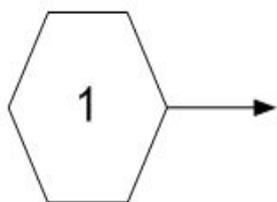
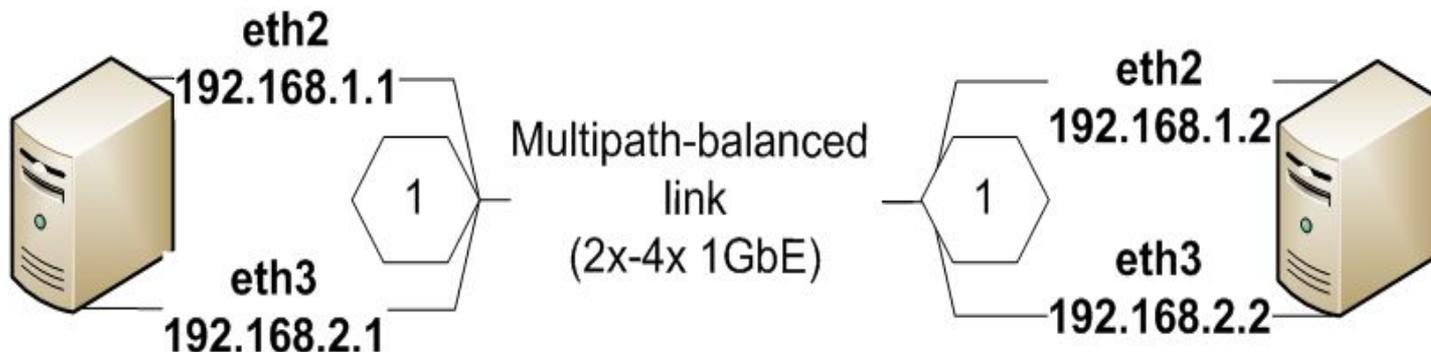
Когда данных стало 200-300Гб в сутки



Вторая попытка масштабирования

- Научились эффективно балансировать нагрузку на сеть (а именно TCP сессии iscsi)
- Учли особенности работы с кластеризованной СУБД
- Проблема: монолитность системы

Вторая попытка масштабирования

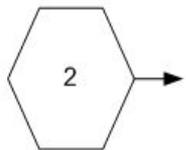
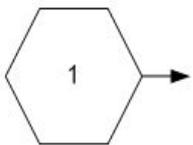


Изменять «путь» доступа к блочному устройству каждые N операций ввода-вывода. Оптимальное значение N между 100 и 1000.

Финальное решение



При отправке каждого пакета инкрементировать счетчик пакетов, рассчитать хеш для поиска исходящего интерфейса как $rx_pkt_count \% if_count$. Отправить пакет через выбранный интерфейс, при отправке подменить source mac address на permanent hwaddr выбранного интерфейса



Для балансировки нагрузки использовать хеширование src-mac или src-dst-mac

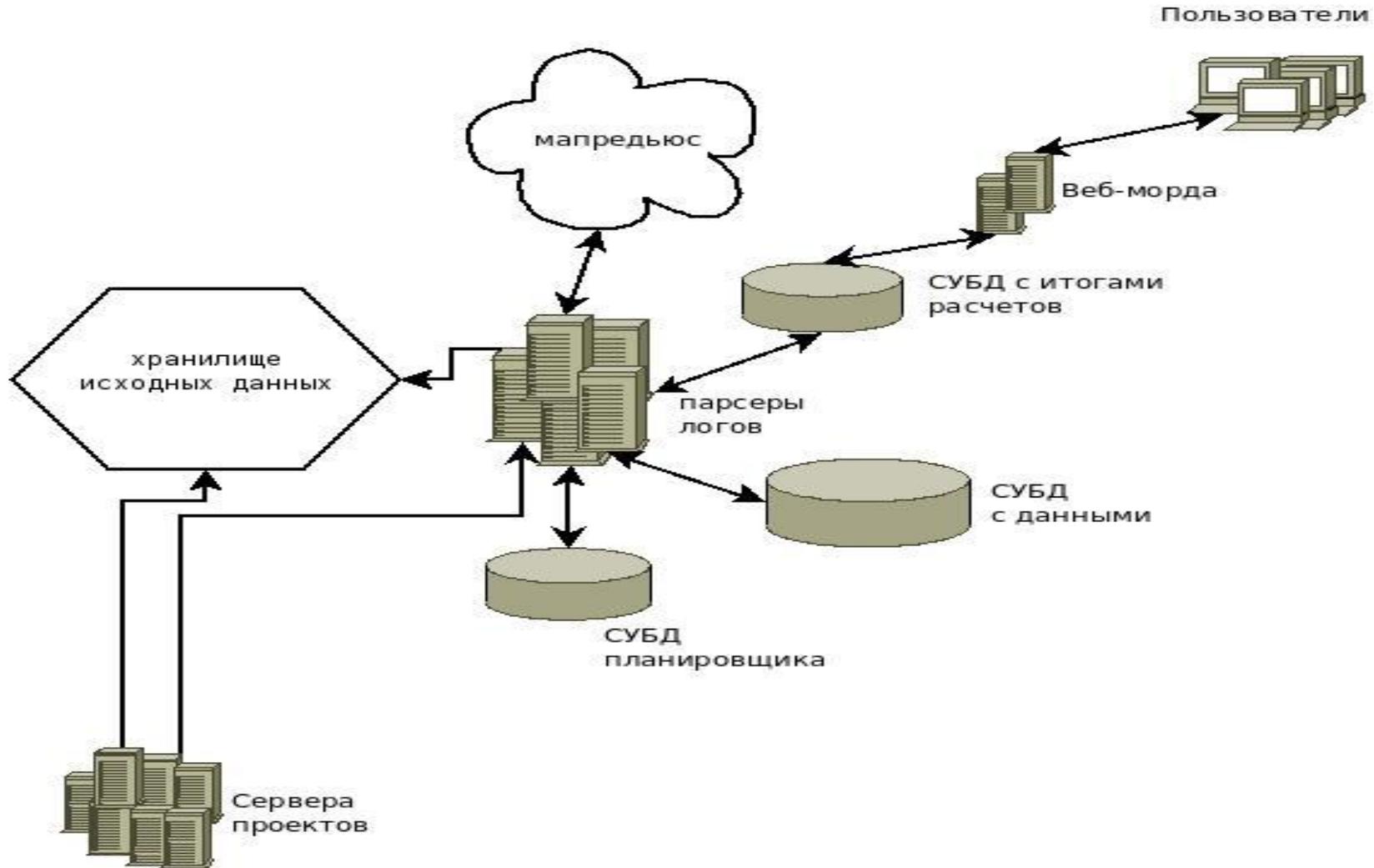


БУРАН

МРІЯ



Сейчас ежесуточно статистика обрабатывает 2-3Тб данных

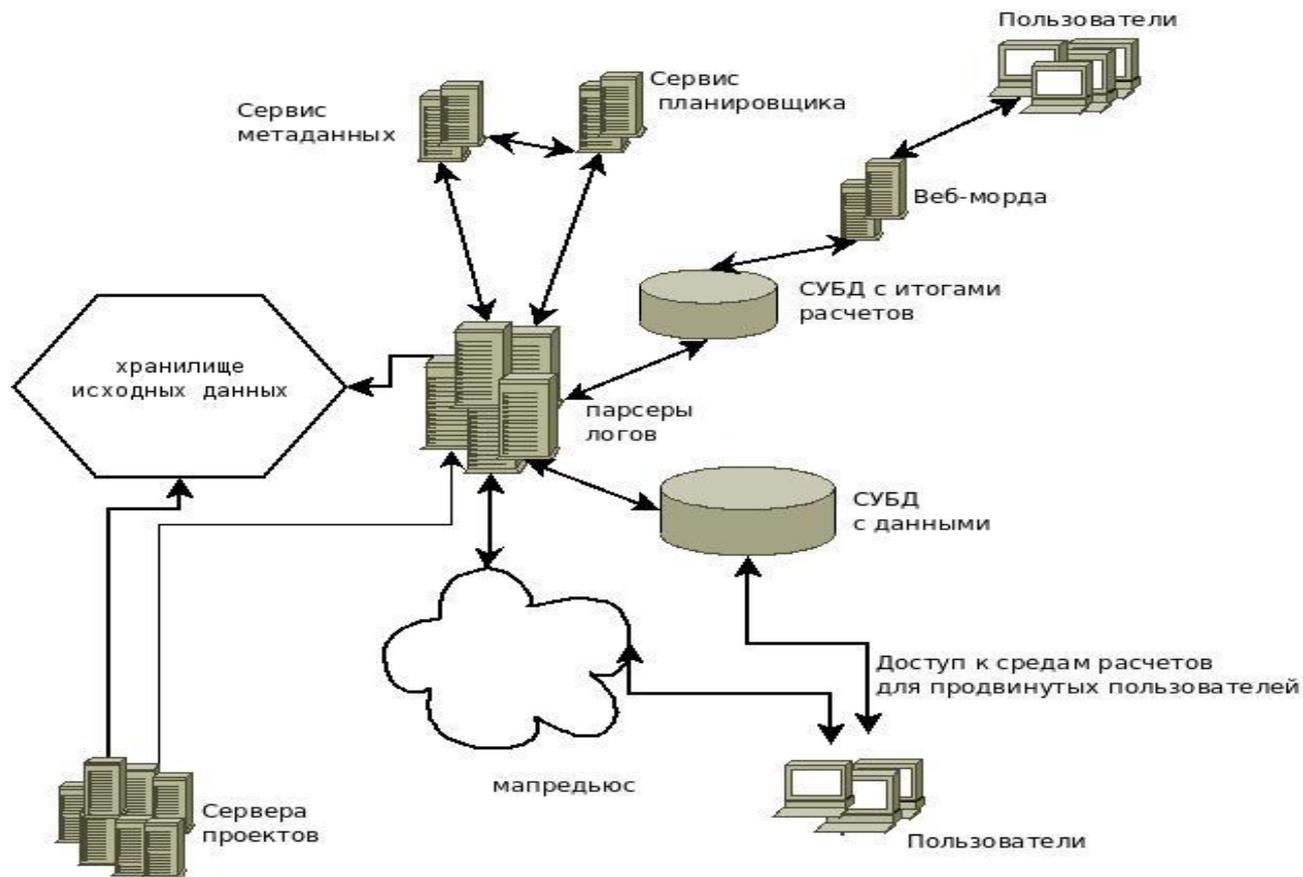




Пересмотр архитектуры

- "оптимальность инструмента" расчетов
- разделение монолитной системы на обособленные функциональному признаку подсистемы

А будет еще больше...



Рост в 3-4 раза каждый год!



Пересмотр архитектуры: почему?

Недостатки старой архитектуры:

- сложности масштабирования
- МОНОЛИТНОСТЬ СИСТЕМЫ

Достоинства старой архитектуры:

- единая точка входа
- SQL-интерфейс
- ну и то что она как-то работает :)

Пересмотр архитектуры: почему?

Недостатки новой архитектуры:

Необходимость локализации расчетов

Достоинства новой архитектуры

Масштабируемость

Надежность

Большая емкость вычислительных и сетевых ресурсов

ИТОГ:

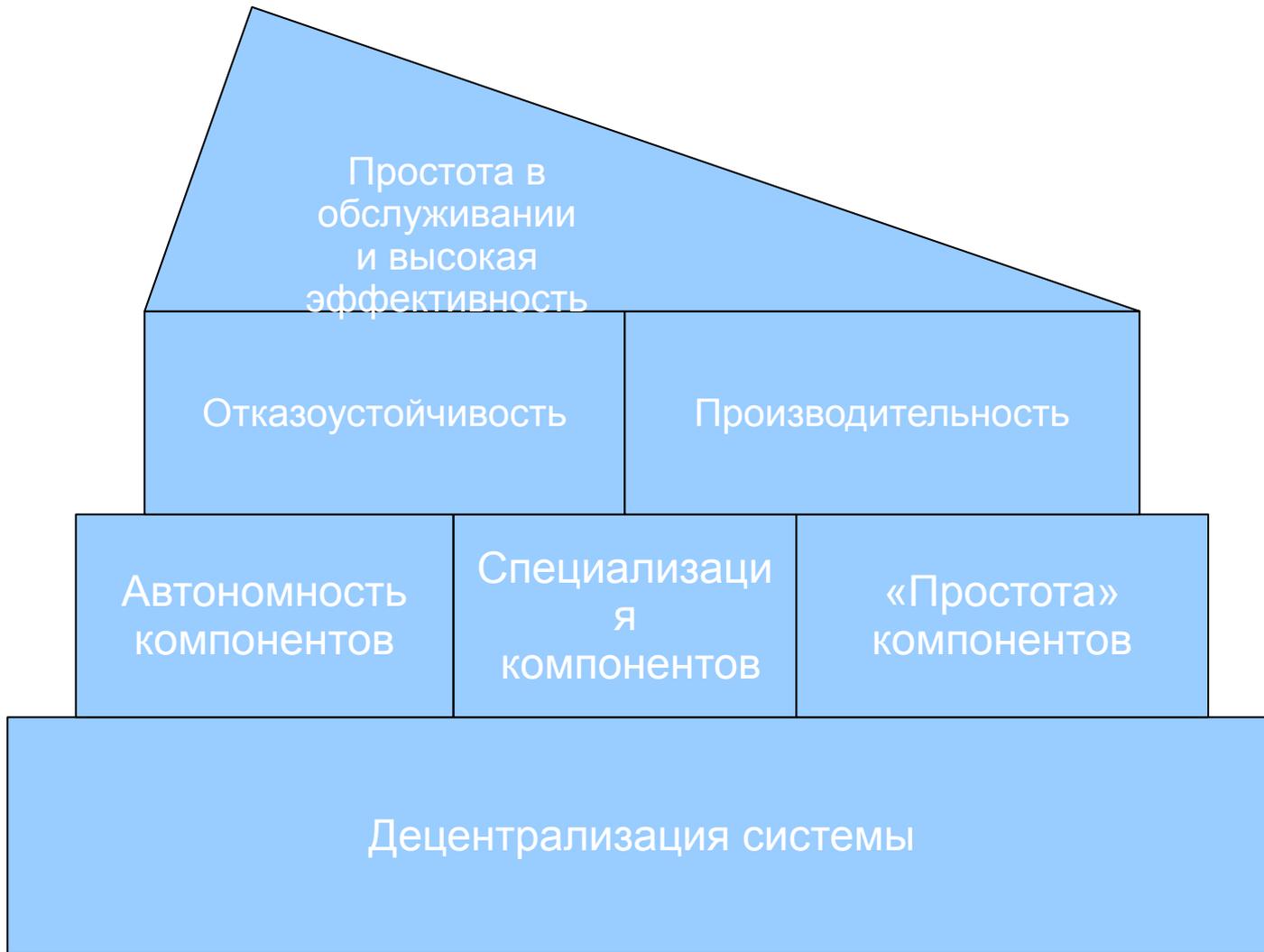
Нерешаемые проблемы решаются просто

Децентрализация системы сильно упрощает ее обслуживание.

Децентрализация системы повышает ее надежность.

(возможно этот слайд нафиг не нужен)

ИТОГ:



Спасибо!

Юхно Олег
руководитель группы администрирования
email: olegu@yandex-team.ru

