Архитектура высоконагруженных проектов.

http://2056.ru

Андрей Светлов

andrew.svetlov@gmail.com
asvetlov.blogspot.com

UA PyCon

Конец октября

2 дня
400 участников

http://ua.pycon.org/

Приглашаем!!!

2056.ru

- Браузерная игра
- Походовая MMORPG
- Свободное перемещение игроков по миру
- Пошаговые боевки
- Города: торговля, задания, изготовление вещей и проч
- Чат, куда ж без него
- Кланы

Особенности

- Постоянное соединение с сервером
- Много одновременных подключенных пользователей
- Минимальное время ответа
- Масштабируемость

- Разумный компромисс между производительностью и надежностью

Продукты

- Python 3
- ZeroMQ
- Redis
- MySQL
- MsgPack
- ZeroGW
- ProcessBoss

- Coffee script
- ECO templates
- less
- JQuery
- Flash

Клиент

```
class Game
    constructor: (@ui, @document) ->
        @connection = new Connection('/websock')
        @connection.onmessage = @message
    message: (event) =>
        try
            cmd = JSON.parse(event.data)
            @do_dispatch(cmd)
        catch e
            console.log(e)
    call: (args...) ->
        try
            @connection.send(JSON.stringify(args))
        catch e
            @connect_error()
```

ZeroGW

- Frontend server для работы по протоколу HTTP включая WebSocket
- Переправляет все запросы в ZeroMQ соединения
- Сообщения в формате json
- Если клиент не поддерживает WebSocket — переключается на Long Polling
- Служебные сообщения: время, приостановка сервера

Синхронизация времени

- Используется только абсолютное GMT время
- Локальные часы часто выставлены неточно

Соединения ZeroGW: output

- Пользователь ↔ компонент
- Компонент выбирается по префиксу запроса
- Имеют connection ID (CID)
- CID ↔ UID
- Указав CID можно послать сообщение любому подключенному пользователю
- Отключение по таймауту, перебрасывание сессии между закладками браузера и т.д.

Output

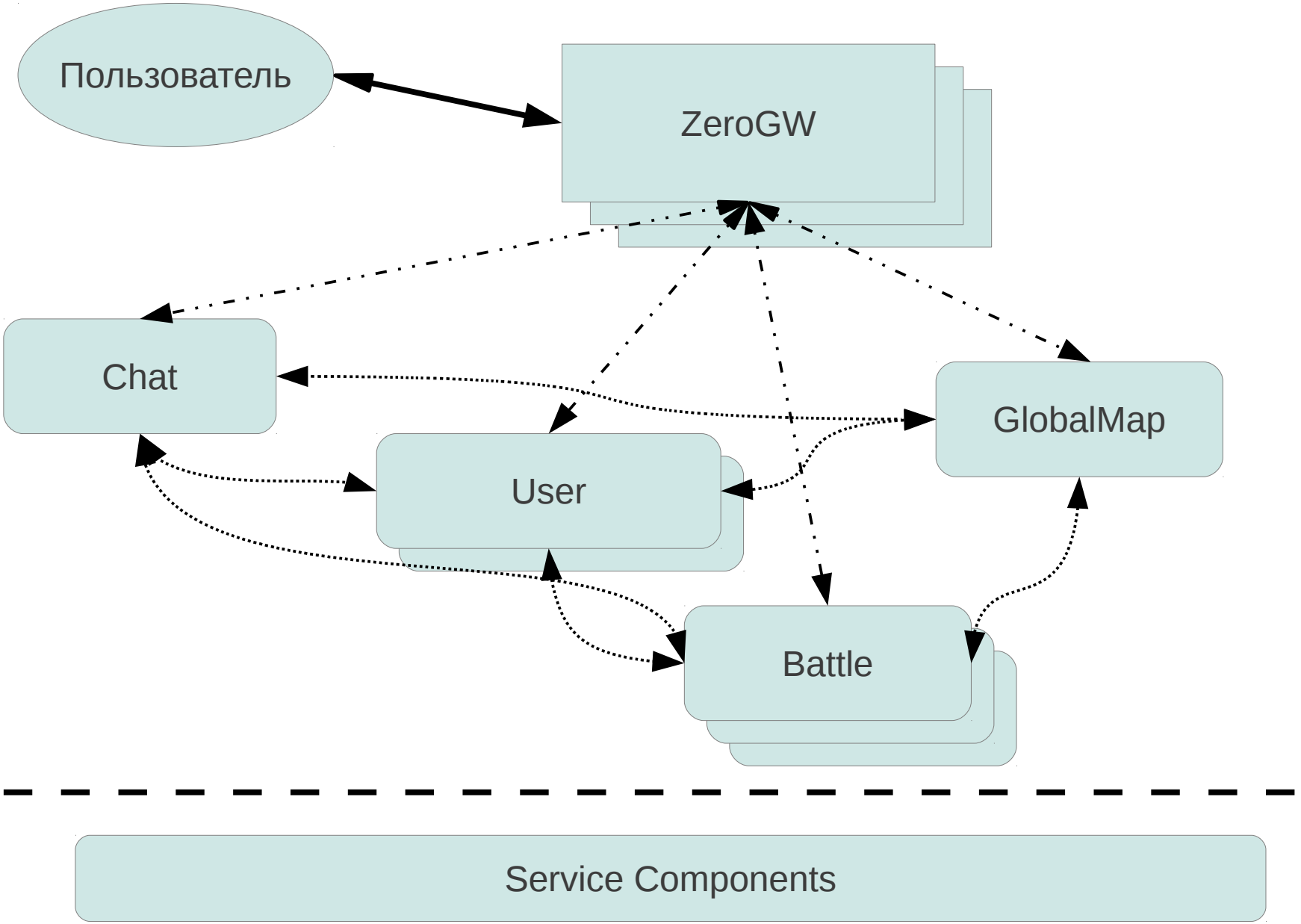
```
web_out.send(b'add_output', cid, b'["home.', b'home_1')
...
encoded = json.dumps(data).encode('ascii')
web_out.send(b'send', cid, encoded)
...
web_out.send(b'drop', cid)
web_out.send(b'del_output', cid)
```

Соединения ZeroGW: topic

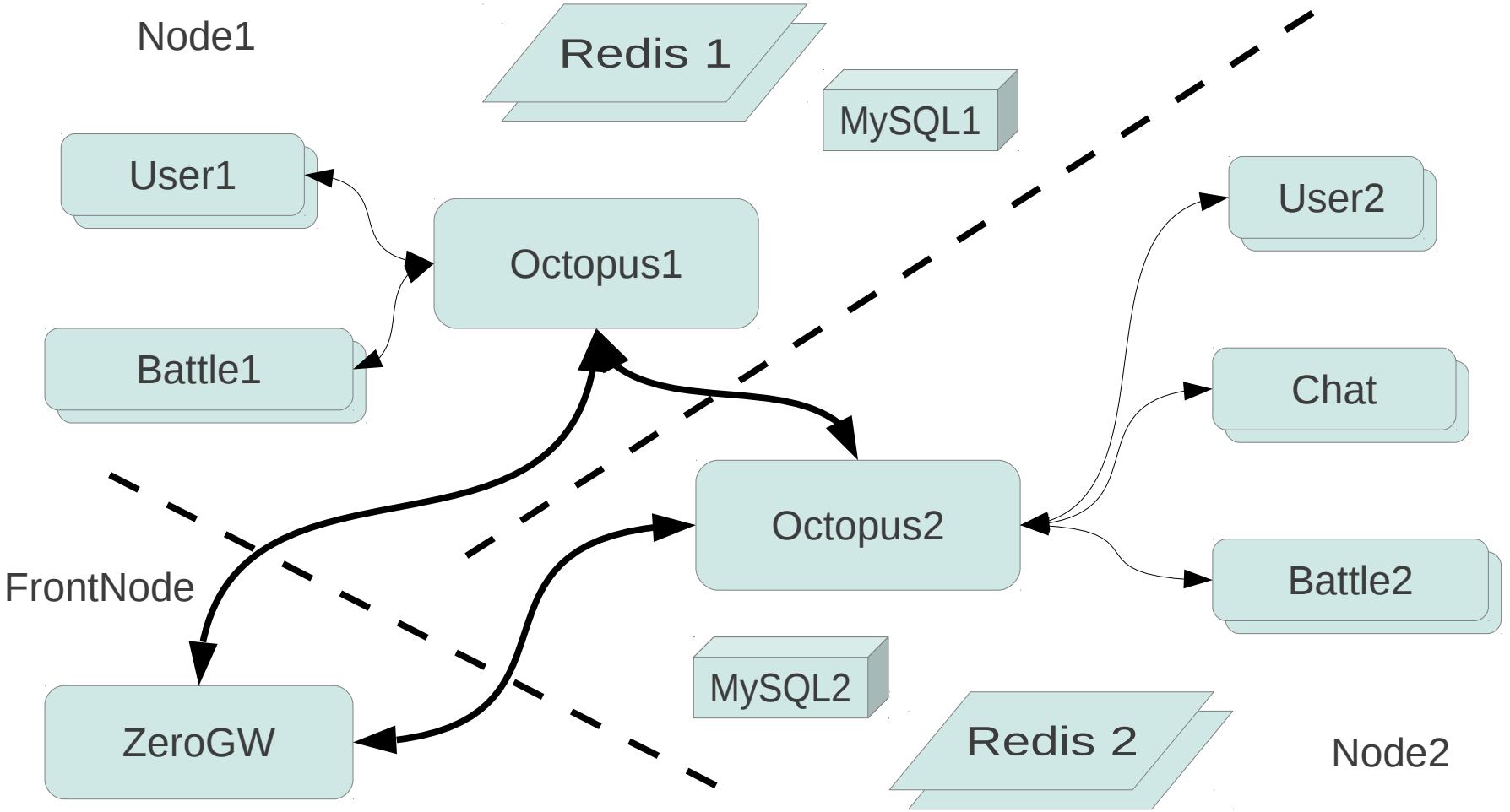
- ID канала → подписанные пользователи
- Служат для широковещательной передачи от сервера клиентам
- Заметно экономят ресурсы

Topic

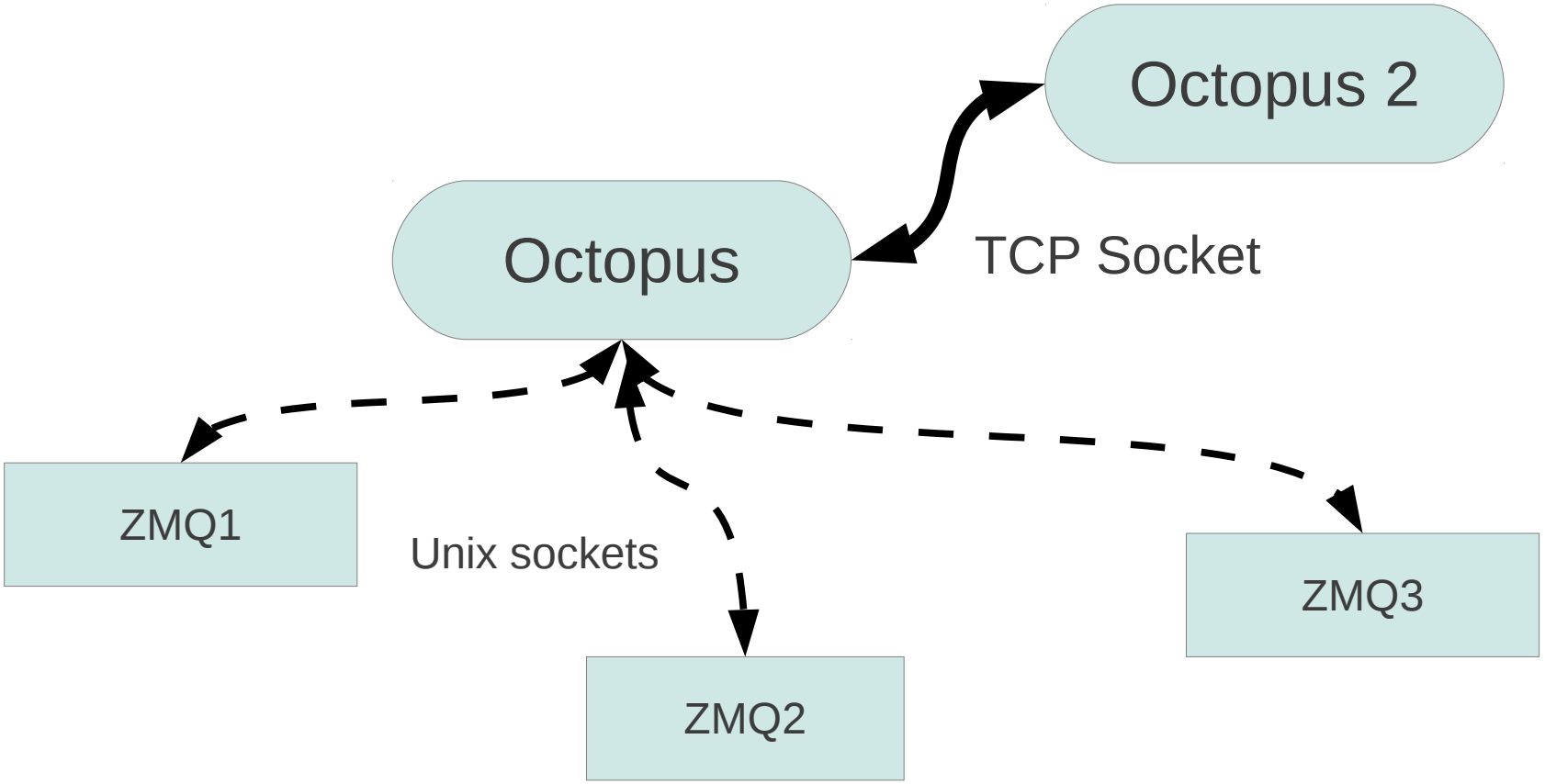
```
web_out.send(b'subscribe', cid, b'chat:room_1')
...
encoded = json.dumps(data).encode('ascii')
web_out.send(b'publish', b'chat:room_1', encoded)
...
web_out.send(b'unsubscribe', cid, b'chat:room_1')
```


Пользователь
ZeroGW
Chat
GlobalMap
User
Battle
Service Components

ZeroMQ Device



Octopus



Компонент

- Отдельный *однопоточный* процесс
- Запускается в нескольких экземплярах
- Легко масштабируется на кластер
- Соединен с клиентом и другими компонентами через ZeroMQ сокеты
- Имеет при необходимости доступ к кешу и БД

Общение компонентов

- RPC (Request-Reply)
- Notify (Push-Pull)
- RPC ожидает ответа, блокируя вызывающую сторону
- RPC (теоретически) может обработать ошибку, случившуюся на другой стороне
- Notify — быстрый, неблокирующий и ненадежный

Типичный компонент

- Ничего не хранит в памяти
- Блокирующе обрабатывает 1 запрос за раз
- Все данные лежат в редисе и БД
- Базы общие для всех однотипных компонентов одной машины, горизонтальное масштабирование (sharding) между узлами кластера
- Смерть процесса ничего не ломает

Боёвка

- Хранит состояние в памяти
- При уничтожении процесса теряются все идущие в нем бои, пользователи выбрасываются туда, откуда пришли
- Не взаимодействует с Редисом/БД

Принцип разделения

- Минимизация потоков данных между разными типами компонентов
- Компонент не видит кеши/базы других компонентов
- Общается только через ZeroMQ каналы
- Шардинг накладывает свои ограничения

Кеширование

- Кешируем всё!!!
- Почти все лежит в msgpack
- Redis отлично справляется, пока находится в памяти
- MySQL используется для «разогрева» кеша, запись сквозная в MySQL и Redis.
- Redis не персистентный.

MySQL

- Шардинг (ручной)
- MyISAM (нетранзактное хранилище)
- Все операции атомарные на уровне SQL запросов.
- MySQL как несколько усложненный вариант key-value storage
- Способен хранить большие объемы данных
- Для сложных структур — JSON

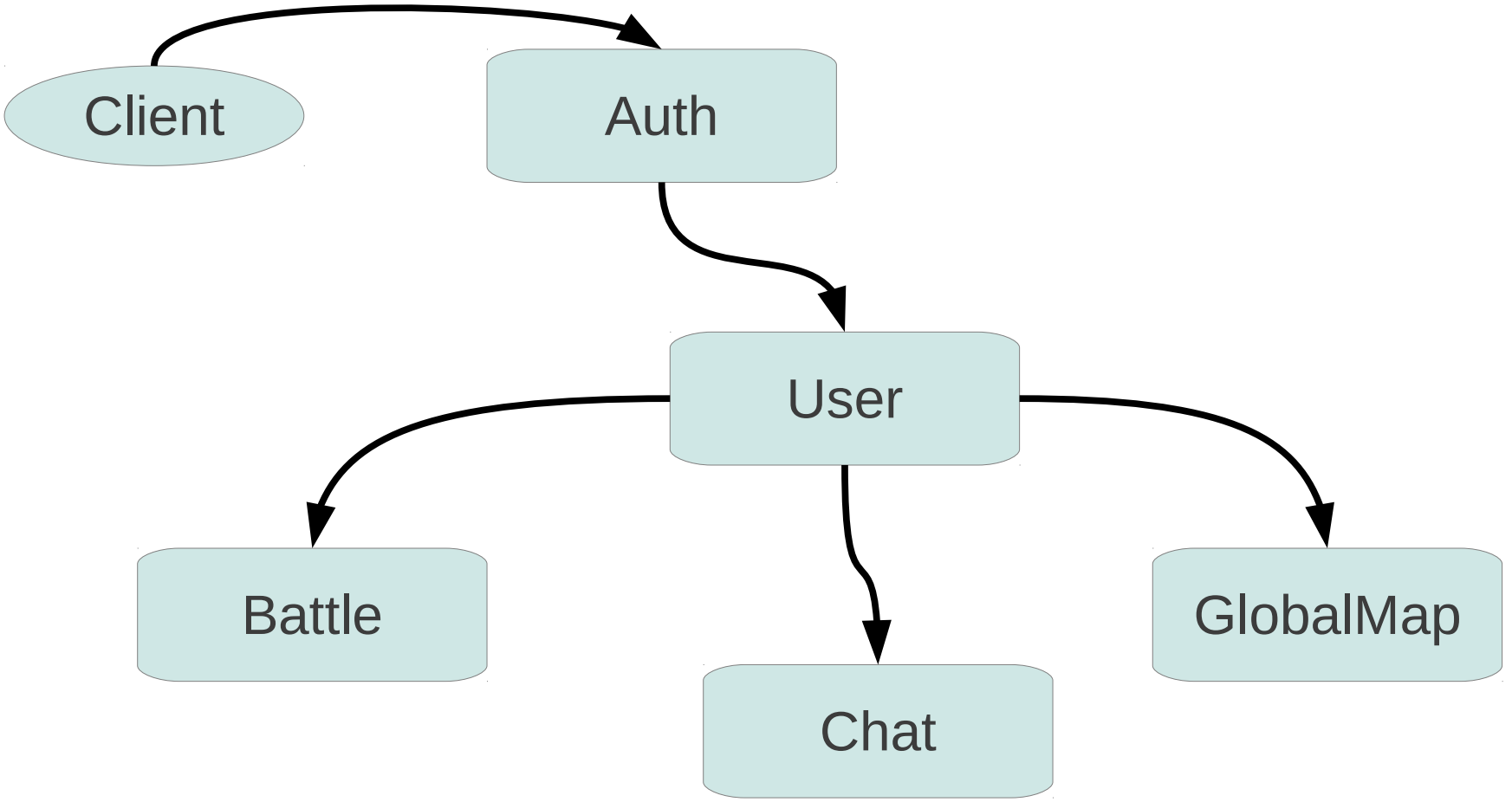
MySQL 2

- Никаких relations
- Только атомарные запросы на изменение
- Полная денормализация
- 15 таблиц на всё

Надежность данных

- Redis всегда можно очистить. Это происходит при каждом обновлении.
- При загрузке невалидных данных из БД они игнорируются, оставаясь в базе + логи.

Login



Reload

- Клиент корректно восстанавливает свое состояние
- При повторном открытии игры последний — победил
- Logout по таймауту
- Новая сессия прописывается во всех компонентах

Логи

- Свой формат бинарного лога (msgpack)
- Пишет сериализованные объекты
- Анализируются статистикой
- Инструмент чтения логов
- Могут быть показаны во множестве различных видов-представлений
- Пишем всё, что происходит в системе

Логи — основа статистики

- При ротации логов файл поступает на анализ
- Все интересующие записи обрабатываются и результат пишется в БД
- Имеем очень много метрик, постоянно создаются новые

Игровой контент

- База данных не лучшее решение
- Yaml — наше всё
- Контент менеджер
- Валидатор
- Генератор статики
- Создавать редактор — напрасно тратить время
- Отчеты и графики

Разработка

- Многопоточный однопроцессный сервер
- Юниттесты
- Функциональные тесты
- Регрессионные тесты
- Проверка контента
- Подсистема моделирования, отчетов и т.д.
- ...

Развертывание

- bossd
  - zerogw
  - redis-namedb
  - redis-user
  - redis-chat
  - **user-1**
  - **user-2**
  - **chat**
  - octopus

Достоинства

- Быстро, весело, экономно
- Высокая нагрузка в пересчете на одного пользователя
- Практически нет архитектурно узких мест — все горизонтально масштабируется

Недостатки

- Приходится думать
- Довольно низкоуровневые интерфейсы (скажем «нет» фреймворкам).
- Необходимость мониторинга и подстраивания кода под распределение нагрузок
- Миграция данных, обращения к другим узлам кластера — стандартные проблемы шардинга
- Забивание ZeroMQ очередей при выходе из строя отдельного компонента

Пузомерка (SLOC)

- Python: 55k
- Python tests: 30k
- Coffee: 31k
- Coffee tests: 10k
- ActionScript: 32k
- Yaml: 84k
- C (OpenSource): 20k
- Art: 2GB

Вопросы?

Архитектура высоконагруженных проектов.
http://2056.ru

Андрей Светлов

andrew.svetlov@gmail.com
asvetlov.blogspot.com