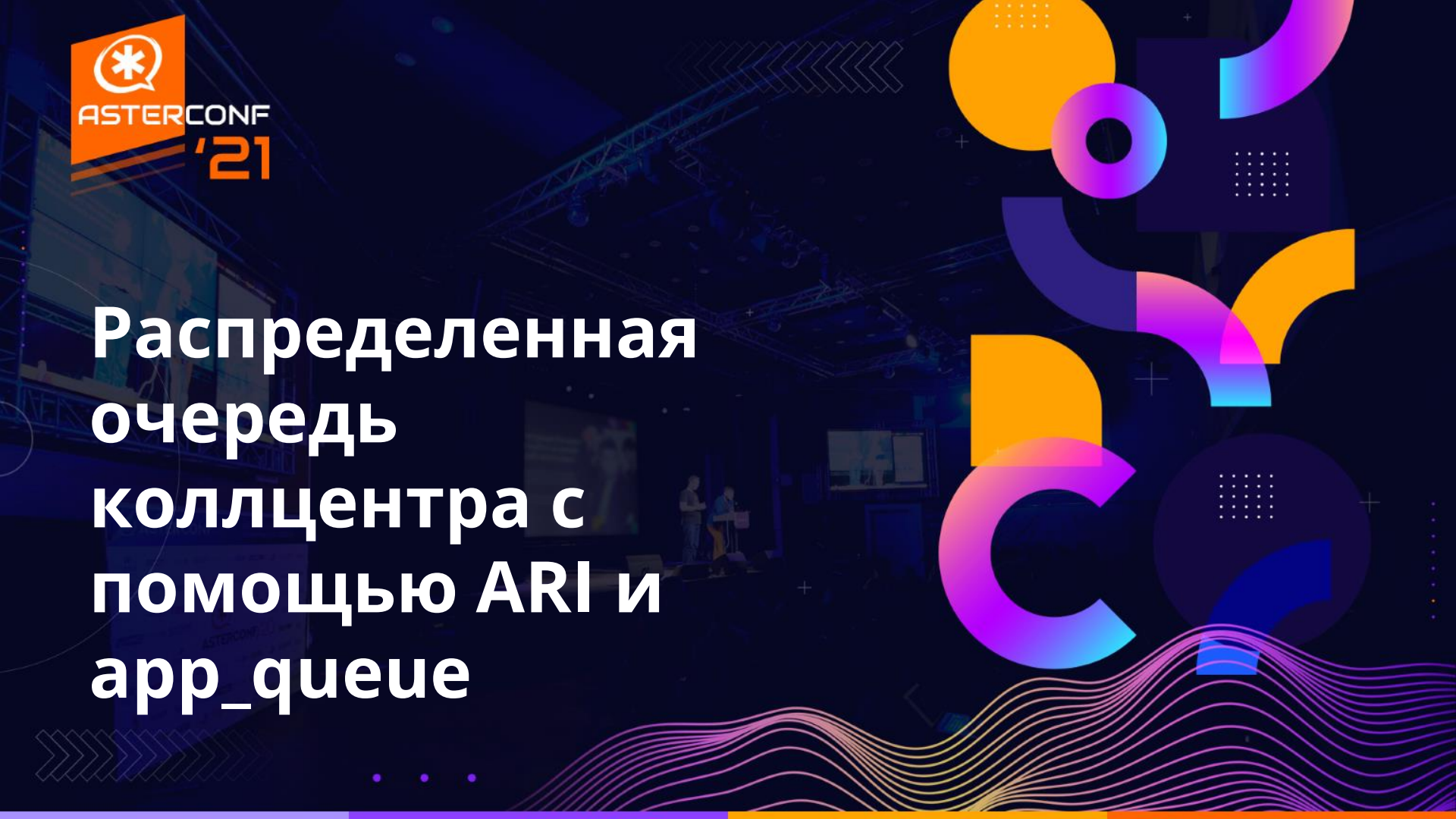
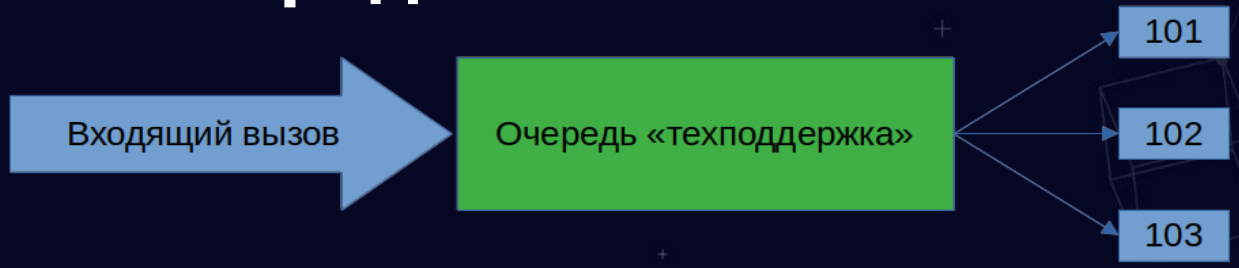
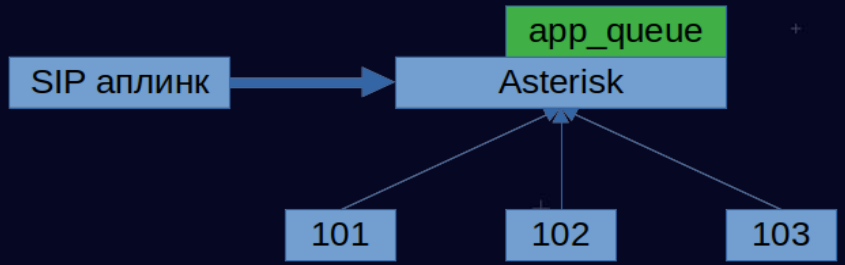


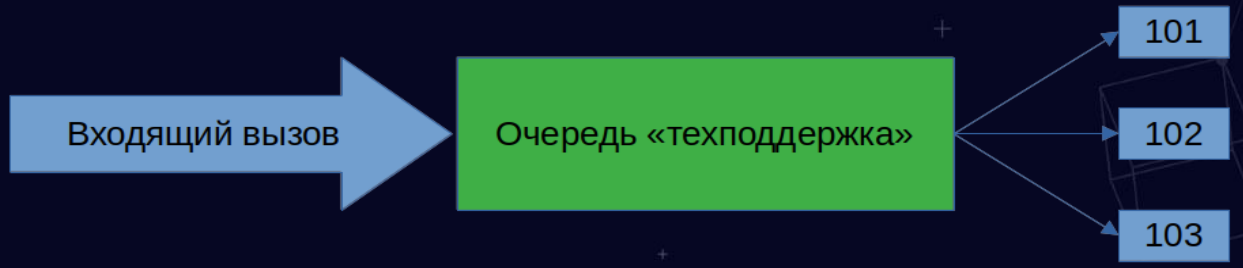
Распределенная очередь коллцентра с помощью ARI и app_queue

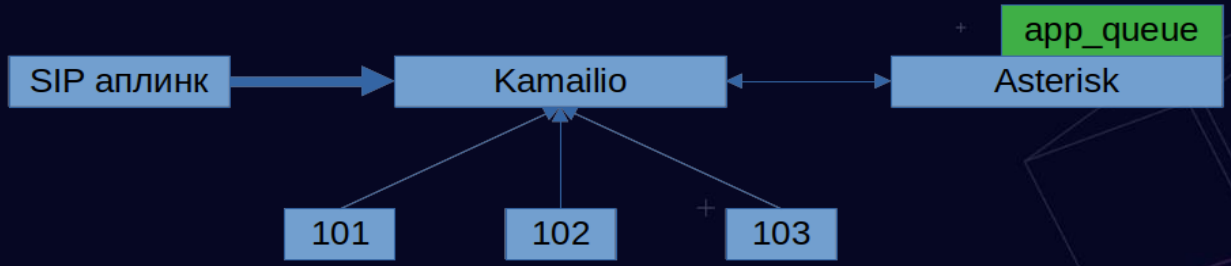
Типовая очередь звонков



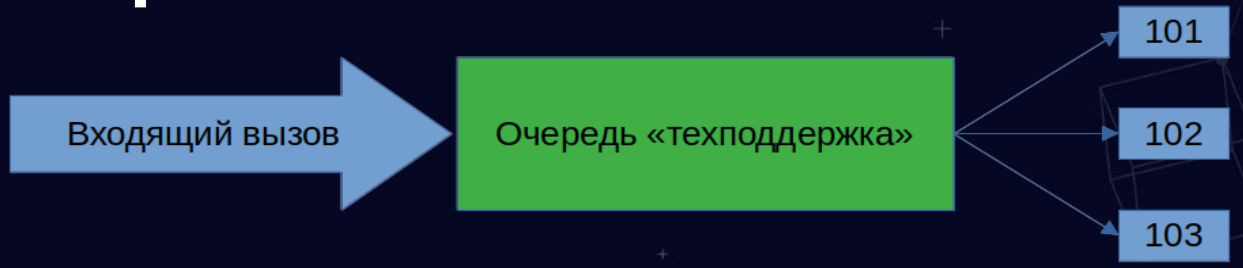


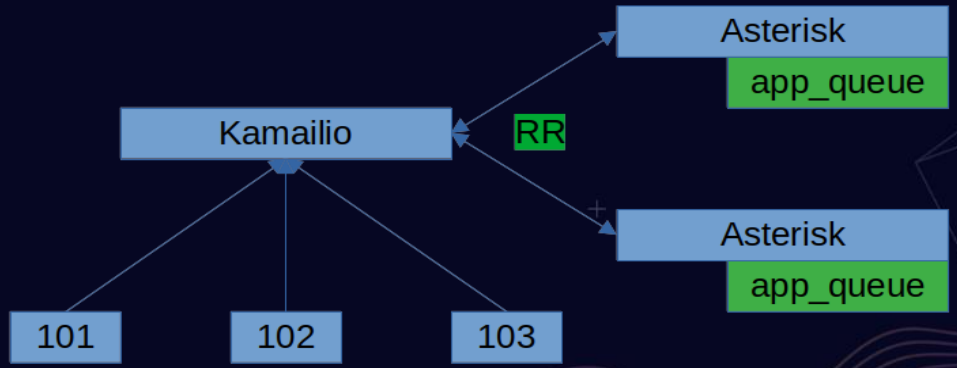
Первые «сложности»





Два астериска





Коллцентр курильщика

Да никак.

Надо выкинуть app_queue и переписать на ARI

Что такое app_queue?

- Очередь

Что такое app_queue?

- Очередь
- Операторы

Что такое app_queue?

- Очередь
- Операторы
- Стратегии вызова

Что такое app_queue?

- Очередь
- Операторы
- Стратегии вызова
- Музыка

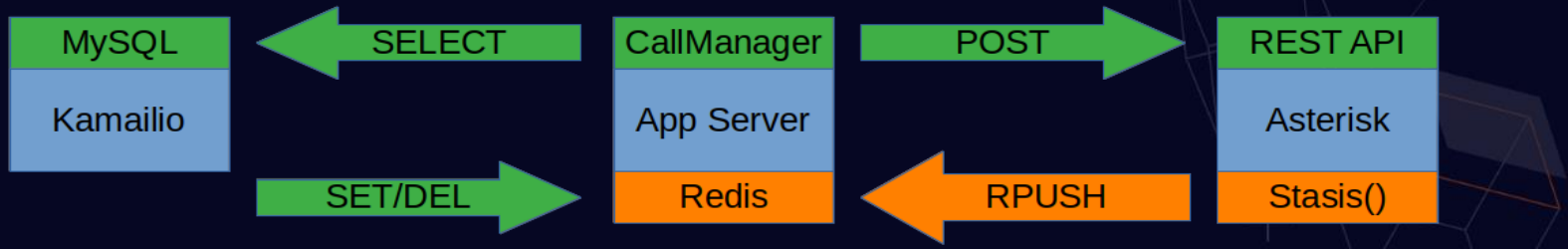
Что такое app_queue?

- Очередь
- Операторы
- Стратегии вызова
- Музыка
- Уведомление о позиции в очереди

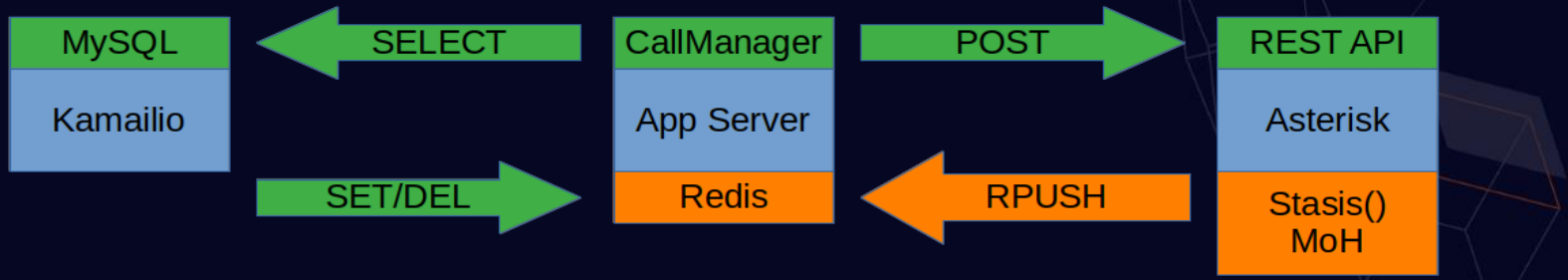
Рисуем архитектуру

- Очередь

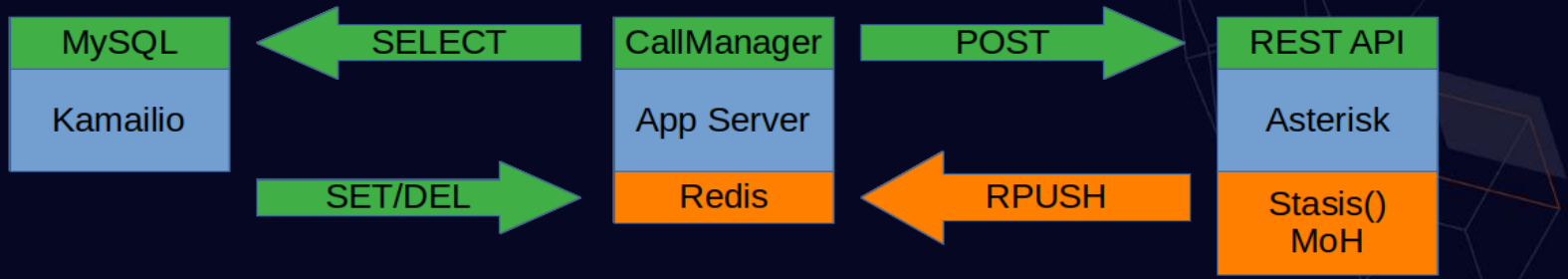
- Очередь
- Операторы
- Стратегии обзвона



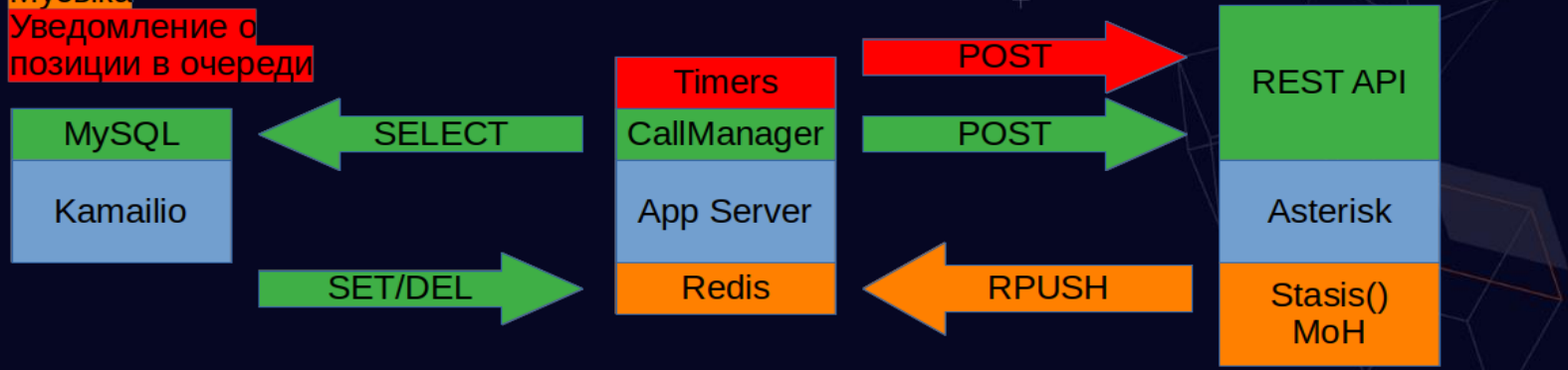
- Очередь
- Операторы
- Стратегии обзвона
- Музыка



- Очередь
- Операторы
- Стратегии обзвона
- Музыка
- Уведомление о позиции в очереди



- Очередь
- Операторы
- Стратегии обзвона
- Музыка
- Уведомление о позиции в очереди



Проектирование модулей

Что из себя представляет очередь в Redis?

- Имя списка, определяемое динамически

Что из себя представляет очередь в Redis?

- Имя списка, определяемое динамически
- Элемент списка содержащий 2 вещи:
  - Идентификатор сервера
  - Имя канала в Stasis

Что из себя представляет очередь в Redis?

- Имя списка, определяемое динамически
- Элемент списка содержащий 2 вещи:
  - Идентификатор сервера
  - Имя канала в Stasis
- Ключ с TTL отвечающий за таймаут нахождения в очереди

Что из себя представляет очередь в Redis?

- Имя списка, определяемое динамически
- Элемент списка содержащий 2 вещи:
  - Идентификатор сервера
  - Имя канала в Stasis
- Ключ с TTL отвечающий за таймаут нахождения в очереди
- Ключ с TTL отвечающий за оповещение

```ruby
Ari.client.on :stasis_start do |e|
  params = e.args.collect { |arg| arg.split('=') }.to_h
  puts "#{Time.now} Received call to #{e.channel.dialplan.exten} in context #{e.channel.dialplan.context}!"

  begin
    queue_name = e.channel.get_channel_var(variable: 'QUEUE_NAME').value
  rescue
    queue_name = "#{e.channel.dialplan.exten}@#{e.channel.dialplan.context}"
    e.channel.set_channel_var(variable: 'QUEUE_NAME', value: queue_name)
  end

  begin
    retries ||= 0
    $redisTimers.set("timeout::#{queue_name}"\
                     "::#{e.channel.id}|#{HOST_ID}", '1', ex: params.fetch('timeout', $CFG['OPTIONS'][:queue_default_timeout]))

    puts "#{Time.now} Add to queue #{queue_name} call #{e.channel.id}|#{HOST_ID}"
    pos = $redisARI.lpush(queue_name, "#{e.channel.id}|#{HOST_ID}")

    $redisTimers.set("announce::#{queue_name}"\
      "::#{e.channel.id}|#{HOST_ID}"\
      "::#{params.fetch('announce', $CFG['OPTIONS'][:queue_default_announce])}"\
      "::#{pos}",\
      "1",\
      ex: 3)
  rescue => ex
    STDERR.puts "#{Time.now} Get exception #{ex.inspect} from #{ex.backtrace}"
    retry if (retries += 1) < 3
  end
  e.channel.start_moh
end
```

```
M.queue = function(context,exten)
  app.NoOp("TEMPLATE. context:"..context.." exten:"..exten.." template:queue");
  app.Ringing();
  app.Answer();
  channel.call_string:set("");
  app.Stasis("QueueARI")
  local call_string = channel.call_string:get();
  if call_string and call_string ~= "" then
    app.NoOp("Call string is set to "..call_string.". Replace TRUNK to "..channel.TRUNK:get());
    _G.dialExtened(string.gsub(call_string, "TRUNK", channel.TRUNK:get()));
  else
    app.NoOp("No call string. Drop call");
  end;
  app.Busy(3);
end;
```

Что из себя представляет CallManager?

- Скрипт, обрабатывающий первый элемент каждого списка в Redis

Что из себя представляет CallManager?

- Скрипт, обрабатывающий первый элемент каждого списка в Redis
- Неизменность данных в Redis*

Что из себя представляет CallManager?

- Скрипт, обрабатывающий первый элемент каждого списка в Redis
- Неизменность данных в Redis*
- MySQL

Что из себя представляет CallManager?

- Скрипт, обрабатывающий первый элемент каждого списка в Redis
- Неизменность данных в Redis*
- MySQL
- Универсальность

```ruby
Daemons.run_proc('callManagerREST', $CFG['OPTIONS']) do
  MYSQL_CFG = loadConfig('mysqlVPBX')
  call_tasks = {}
  @randomizer = Random.new
  begin
    @redis = Redis.new($CFG['REDIS_ARI'])
    @credis = Redis.new($CFG['REDIS_CACHE'])
    @mysql = Mysql2::Client.new(MYSQL_CFG['COMMON'].merge(MYSQL_CFG['RW']))
    @semaphore = Mutex.new
    @script_lock = Redis_lock.new($CFG['REDIS_CACHE'])
  rescue => e
    warn "#{Time.now} Get exception #{e.inspect} from #{e.backtrace}"
    exit(false)
  end

  while @redis && @mysql
    if @script_lock.check_lock("lock::#{$CFG['OPTIONS'][:app_name]}")
      @redis.keys.each do |task|
        next if call_tasks[task]&.alive?

        call_tasks[task] = Thread.new {
          call_agents = get_agents_forcall(get_task_agents(task), get_ring_strategy(task), task)
          sendMSG(call_agents, task) if call_agents && (call_agents != ',') && !call_agents.empty?
        }
      end
    end
    sleep 3
  end
  exit false
end
```

```
1 •  SELECT * FROM vpbx_is74_ru.callManager;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

#	id	taskID	extens	ringStrategy	ringTime	lastCalled
21	57	Polimedica@hotline.vpbx.is74.ru	2103	ringall	360	NULL
22	58	OKB_3@hotline.vpbx.is74.ru	2104	ringall	360	NULL
23	59	Volunteers@hotline.vpbx.is74.ru	2201,2202	ringall	360	NULL
24	60	Hotline-incoming@hotline.vpbx.is74.ru	1010,1011,1012,1013,...	ringall	360	NULL
25	63	2477848@DomainB.vpbx.is74.ru@1	1000	ringall	120	NULL
26	66	2477848@DomainB.vpbx.is74.ru@2	1000,1001	ringall	600	NULL
*	NULL	NULL	NULL	NULL	NULL	NULL

API для управления каналами

- Унификация общения между модулями

API для управления каналами

- Унификация общения между модулями
- Расширяемость

API для управления каналами

- Унификация общения между модулями
- Расширяемость
- Отказоустойчивость

API для управления каналами

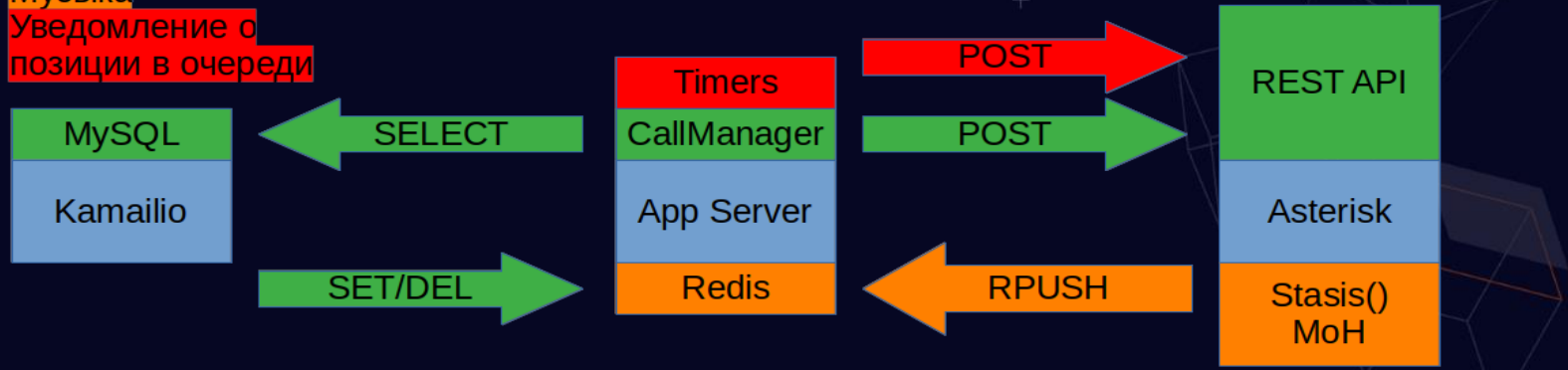
- Унификация общения между модулями
- Расширяемость
- Отказоустойчивость
- Использование не только Asterisk

```ruby
Ari.client = Ari::Client.new($CFG['ARI'])
def call(env)
  headers = {}
  env.select { |k, v| k.start_with? 'HTTP_' }…
  begin
    if env['REQUEST_METHOD'] == 'POST'
      case headers['ACTION']      You, 9 months ago • Навел порядок в коде согласно спеллчекеру
      when 'processCallTask'
        puts "#{Time.now} Set for channel #{headers['CHANNEL']} #{headers['VARNAME']}=#{headers['VARCONTENT']}:"\
             "#{Ari::Channel.set_channel_var(channelId: headers['CHANNEL'], variable: headers['VARNAME'], value: headers['VARCON
        puts "#{Time.now} Continue in dialplan:#{Ari::Channel.continueInDialplan(channelId: headers['CHANNEL'])}"
      when 'announcePosition'
        if headers['POSITION'] == '1'
          puts "#{Time.now} Play announce:#{Ari::Channel.play(channelId: headers['CHANNEL'], media: 'sound:queue-youarenext')}"
        else
          puts "#{Time.now} Play announce:#{Ari::Channel.play(channelId: headers['CHANNEL'], media: 'sound:queue-periodic-announce'
          puts "#{Time.now} Play announce:#{Ari::Channel.play(channelId: headers['CHANNEL'], media: 'sound:queue-thereare')}"
          puts "#{Time.now} Play position:#{Ari::Channel.play(channelId: headers['CHANNEL'], media: "number:#{headers["POSITION"]}"
        end
        puts "#{Time.now} Restart moh:#{Ari::Channel.start_moh(channelId: headers['CHANNEL'])}"
      when 'timeout'
        puts "#{Time.now} Play announce:#{Ari::Channel.play(channelId: headers['CHANNEL'], media: 'sound:sorry')}"
        puts "#{Time.now} Play announce:#{Ari::Channel.play(channelId: headers['CHANNEL'], media: 'sound:tt-allbusy_short')}"
        puts "#{Time.now} Play announce:#{Ari::Channel.play(channelId: headers['CHANNEL'], media: 'sound:pls-try-call-later')}"
        puts "#{Time.now} app_stasis timeout. Continue in dialplan:#{Ari::Channel.continueInDialplan(channelId: headers['CHANNEL'])
      when 'checkChannel'…
      end
    end
    response || [200, { 'Content-Type' => 'text/html' }, ['OK']]
  rescue => e
    case e.message
    when 'Channel not in Stasis application', 'Channel not found'
      puts "#{Time.now} Get '#{e.message}' when work with #{headers['ACTION']} for channel #{headers['CHANNEL']}"
      [404, { 'Content-Type' => 'text/html' }, [e.message]]
```

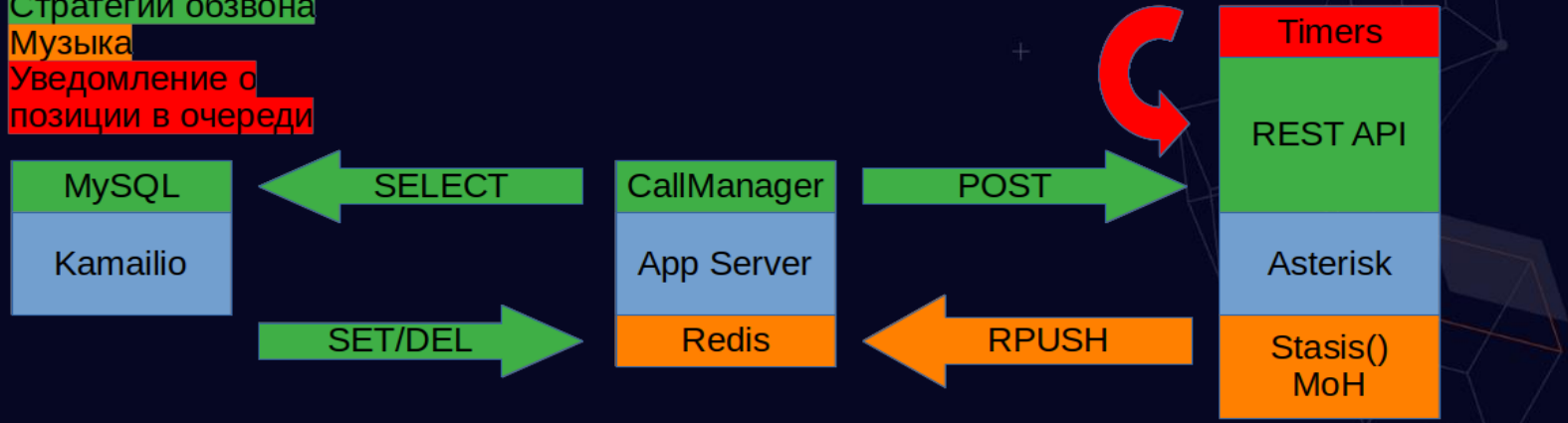
Результаты

- Получилась микросервисная архитектура

- Очередь
- Операторы
- Стратегии обзвона
- Музыка
- Уведомление о позиции в очереди



- Очередь
- Операторы
- Стратегии обзвона
- Музыка
- Уведомление о позиции в очереди



Результаты

- Получилась микросервисная архитектура
- Получилось контейнеризируемое решение

Результаты

- Получилась микросервисная архитектура
- Получилось контейнеризируемое решение
- Получилось масштабируемое решение

Результаты

- Получилась микросервисная архитектура
- Получилось контейнеризируемое решение
- Получилось масштабируемое решение
- Получилось отказоустойчивое решение

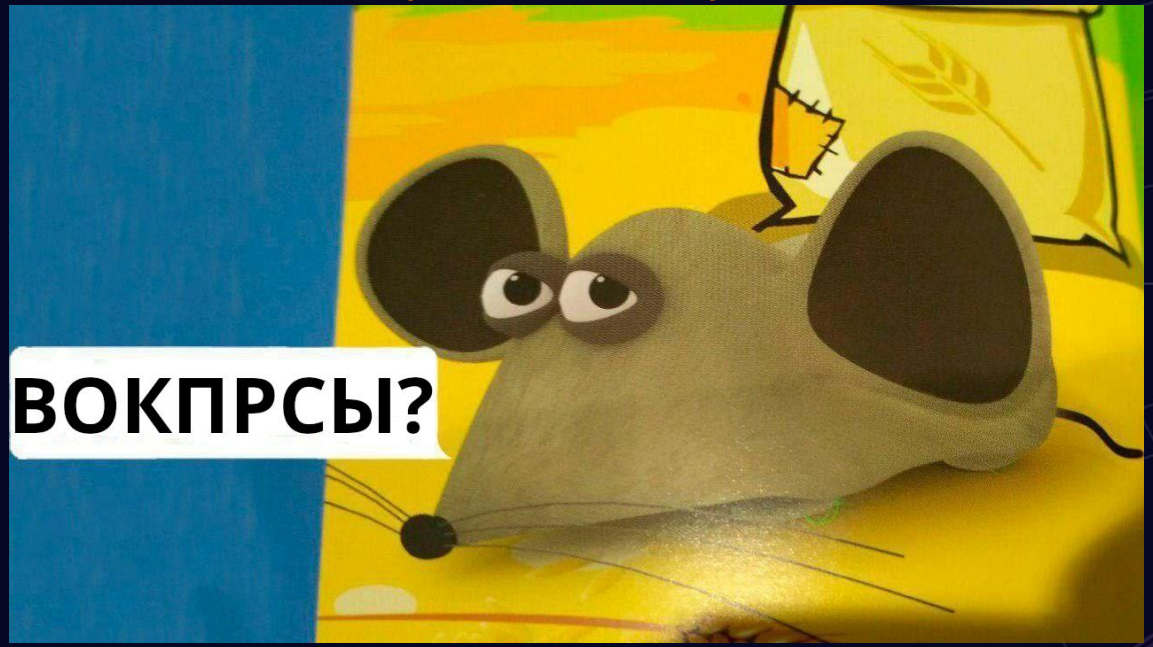
Результаты

- Получилась микросервисная архитектура
- Получилось контейнеризируемое решение
- Получилось масштабируемое решение
- Получилось отказоустойчивое решение
- Получилось универсальное решение - для звонков на группы потребовалось доработать только ARI модуль

Самое главное

Распределенная очередь реализована и работает

Спасибо за внимание!



yarin-aa@intersvyaz.net
8 800 2000 747
is74.ru