

Как приготовить микросервис к продакшену

Вадим Мадисон, М-Тех
2016



<http://www.devconf.ru>



- Услуги системной интеграции, начиная от разработки технологических решений и заканчивая доставкой видео-сигнала до конечного пользователя
- Интернет-видеоплатформа для каналов МатчТВ и НТВ-ПЛЮС
- 300 000 одновременных пользователей
- > 1 000 000 уникальных посетителей в сутки
- Отдаем контента до 300 Тб/час

Кто не знает, что такое
микросервис?

Есть плюсы ...

Плюсы

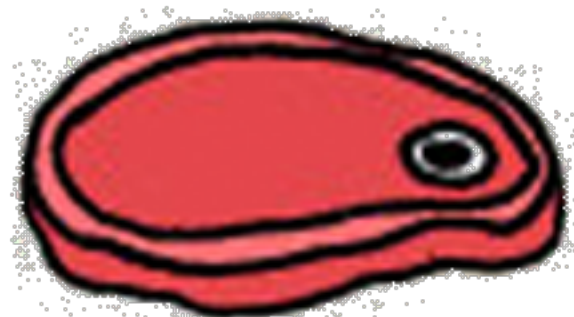
- Независимая разработка
- Лучший контроль технического и технологического долга
- Скорость тестирования, доставки, запуска
- Упрощение масштабирования

ЕСТЬ МИНУСЫ ...

Минусы

- Распределенные коммуникации, дополнительное API, больше точек отказа и логики их обработки
- Оверхед: сетевой, сериализация/десериализация
- Обнаружение сервисов
- Инструментарий!

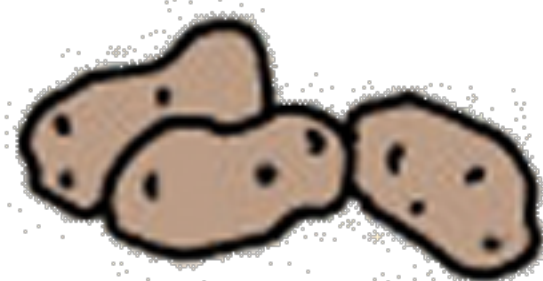
Ингредиенты



мясо



лук



картошка



капуста



свекла



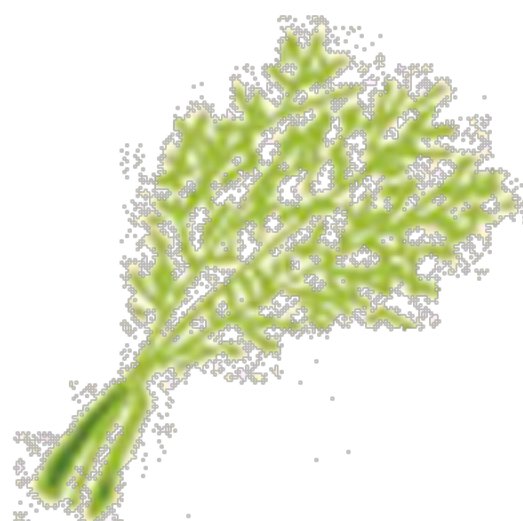
морковь



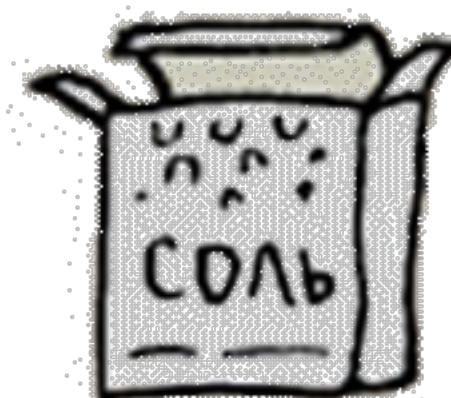
замоченная
фасоль



томат



укроп



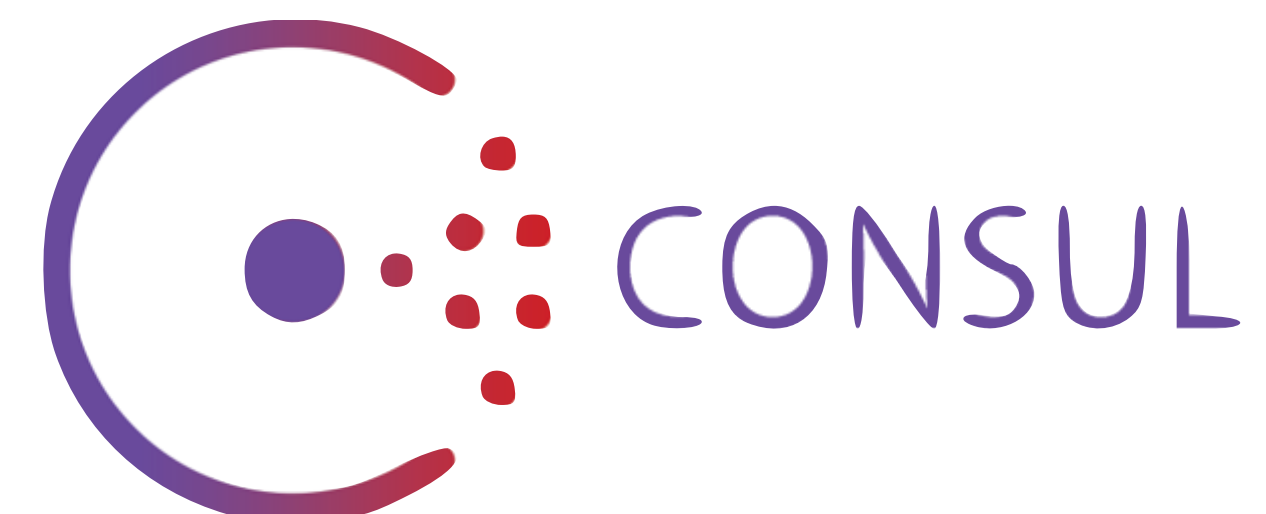
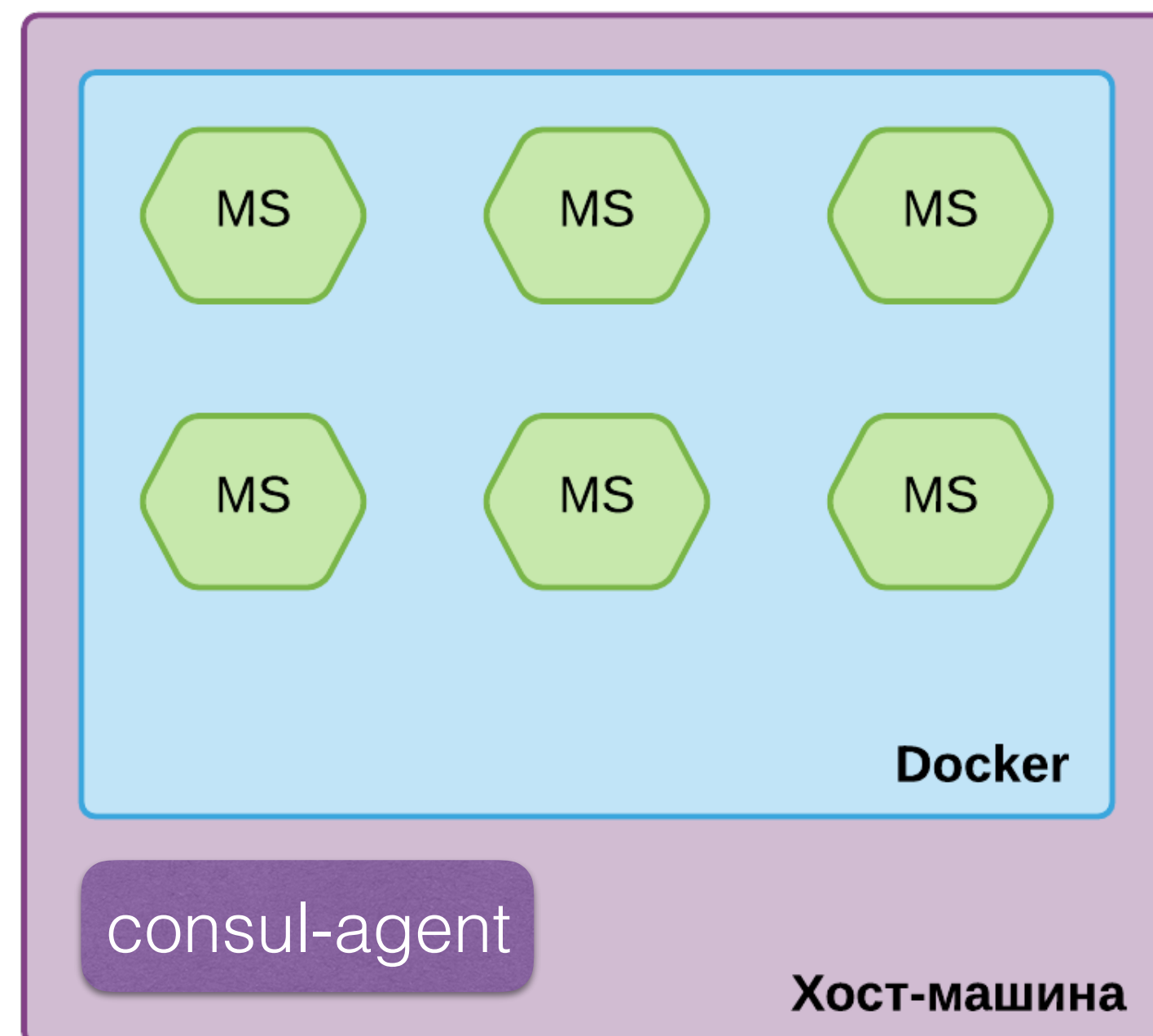
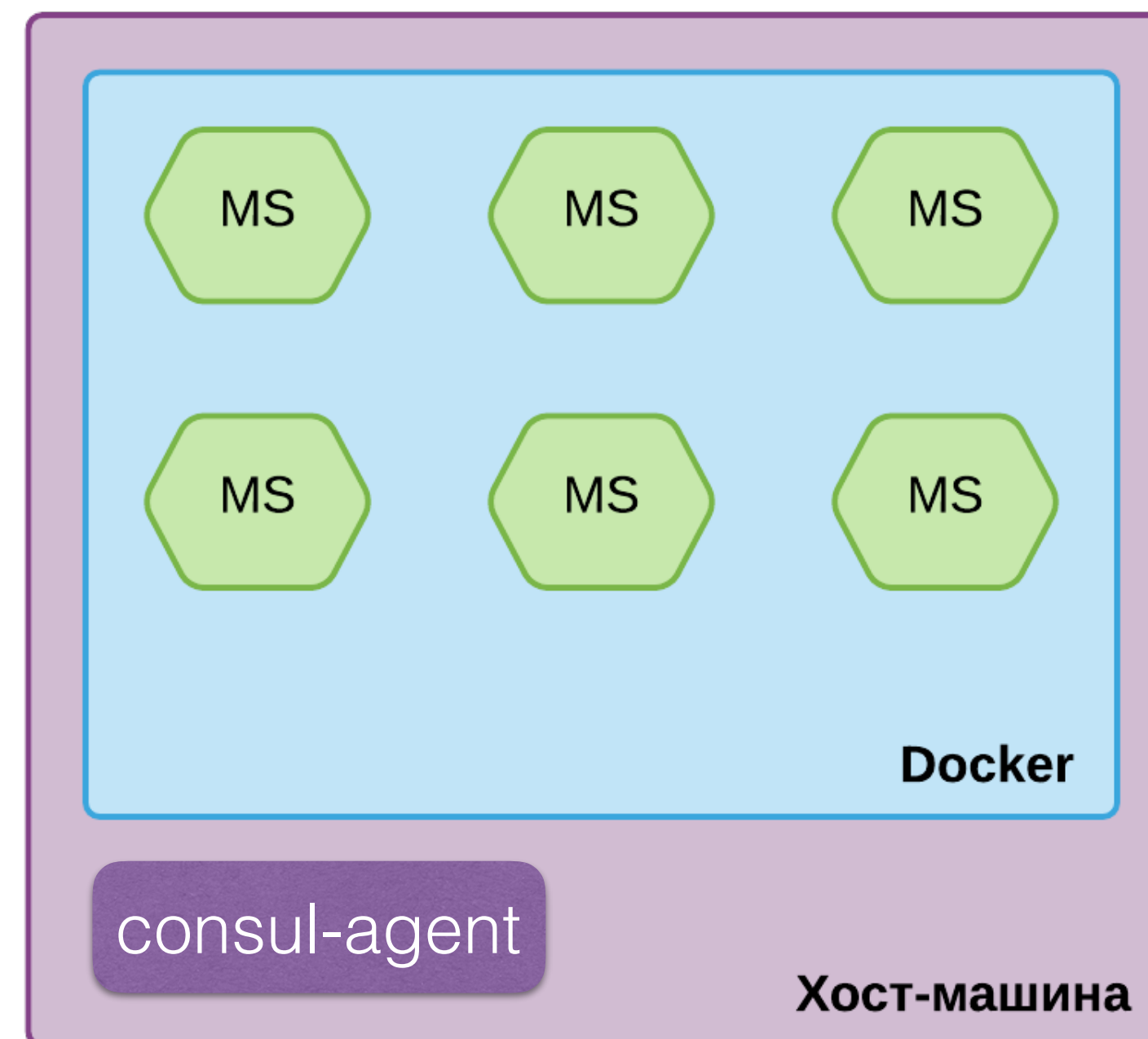
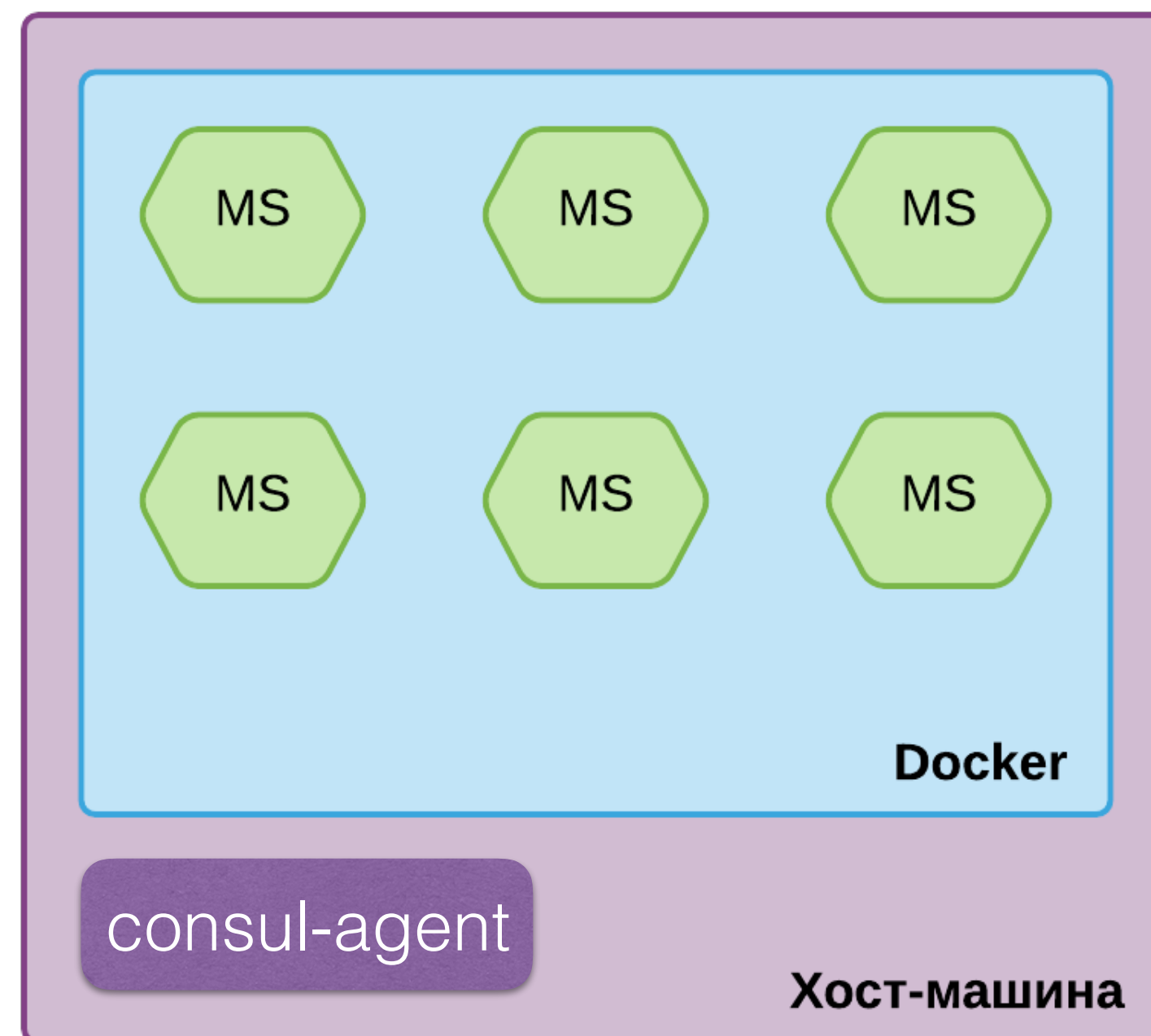
соль



чеснок



сметана



Как ГОТОВИТЬ?

Общие рекомендации

- Не использовать фреймворки

Общие рекомендации

- Не использовать фреймворки
- Собирать проект из шаблона

Конфигурирование



мясо



лук



капуста



картофель



свекла



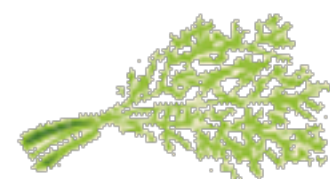
морковь



фасоль
замоченная



х3
томат



укроп



соль



чеснок



сметана



вскипятить

Конфигурирование

- ✗ Переменные окружения (12-факторное приложение)
<http://12factor.net/ru/config>

Конфигурирование

- ✗ Переменные окружения (12-факторное приложение)
<http://12factor.net/ru/config>
- ✗ Не структурированный конфиг
viper.`Get("name")` // this would be "steve"

Конфигурирование

- ✗ Переменные окружения (12-факторное приложение)
<http://12factor.net/ru/config>
- ✗ Не структурированный конфиг
viper.`Get("name")` // this would be "steve"
- ✓ Zero configuration
<https://github.com/hashicorp/consul/tree/master/api>
- ✓ Не храним единым JSON, каждое значение отдельно в KV

Раздел конфигурации

```
type MongoDB struct {  
    Hosts      string  
    DB         string  
    Mode       string  
    ReplicaSet string  
}
```

Загрузка:

```
func (m *MongoDB) LoadFromConsul( ... ) error { ... }
```

Валидация:

```
func (m *MongoDB) validate() error { ... }
```

Конфигурация

```
type Config struct {  
    Mongo      *MongoDB  
    Redis      *Redis  
    Metrics    *Metrics  
    ...  
    TLS        *TLS  
    Log        *Logger  
}
```

Загрузка:

```
func (c * Config) LoadFromConsul( ... ) error { ... }
```

Журналирование



Журналирование

- ✓ Журналирование (12-факторное приложение)
<http://12factor.net/ru/logs>

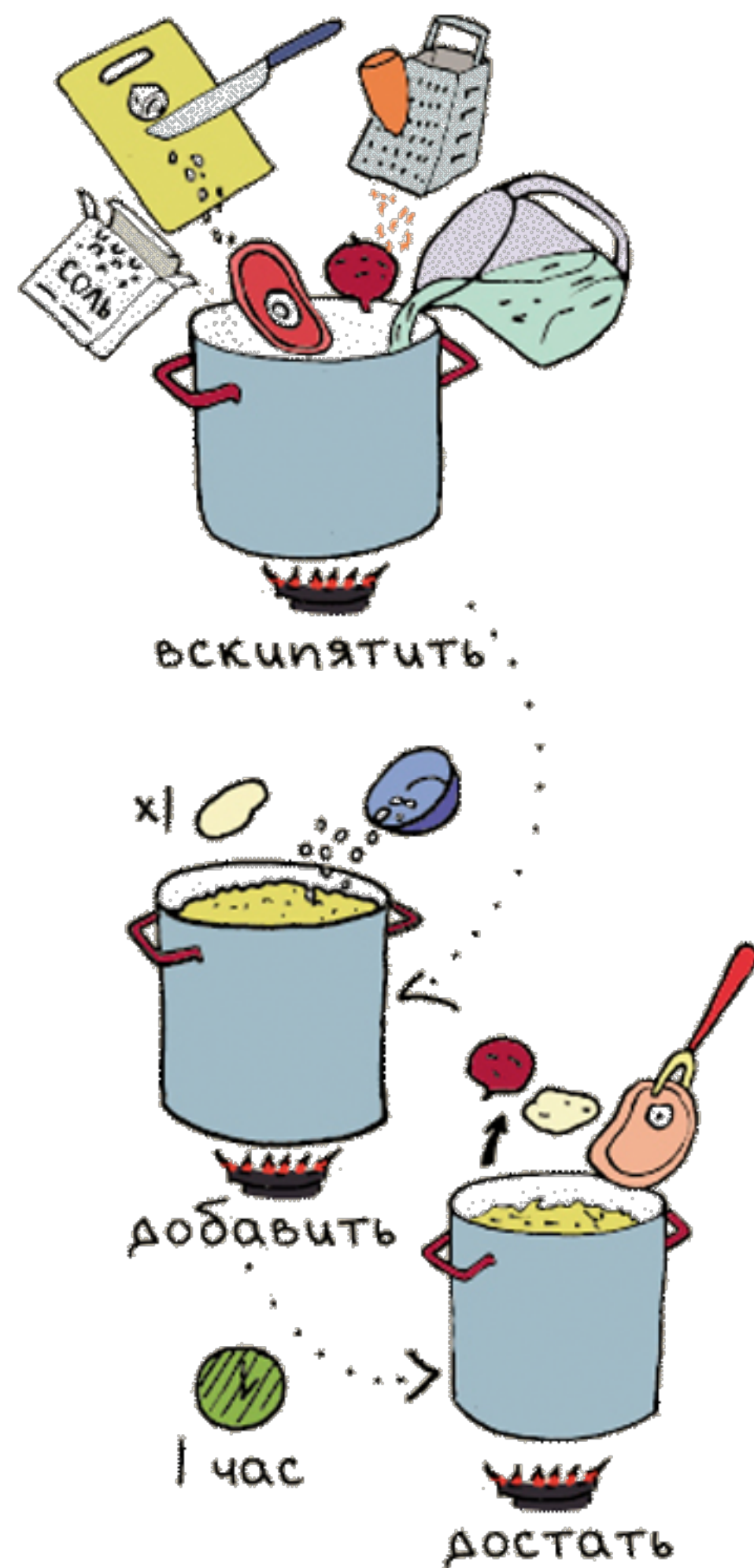
Журналирование

- ✓ Журналирование (12-факторное приложение)
<http://12factor.net/ru/logs>
- ✗ Logrus
<https://github.com/Sirupsen/logrus>

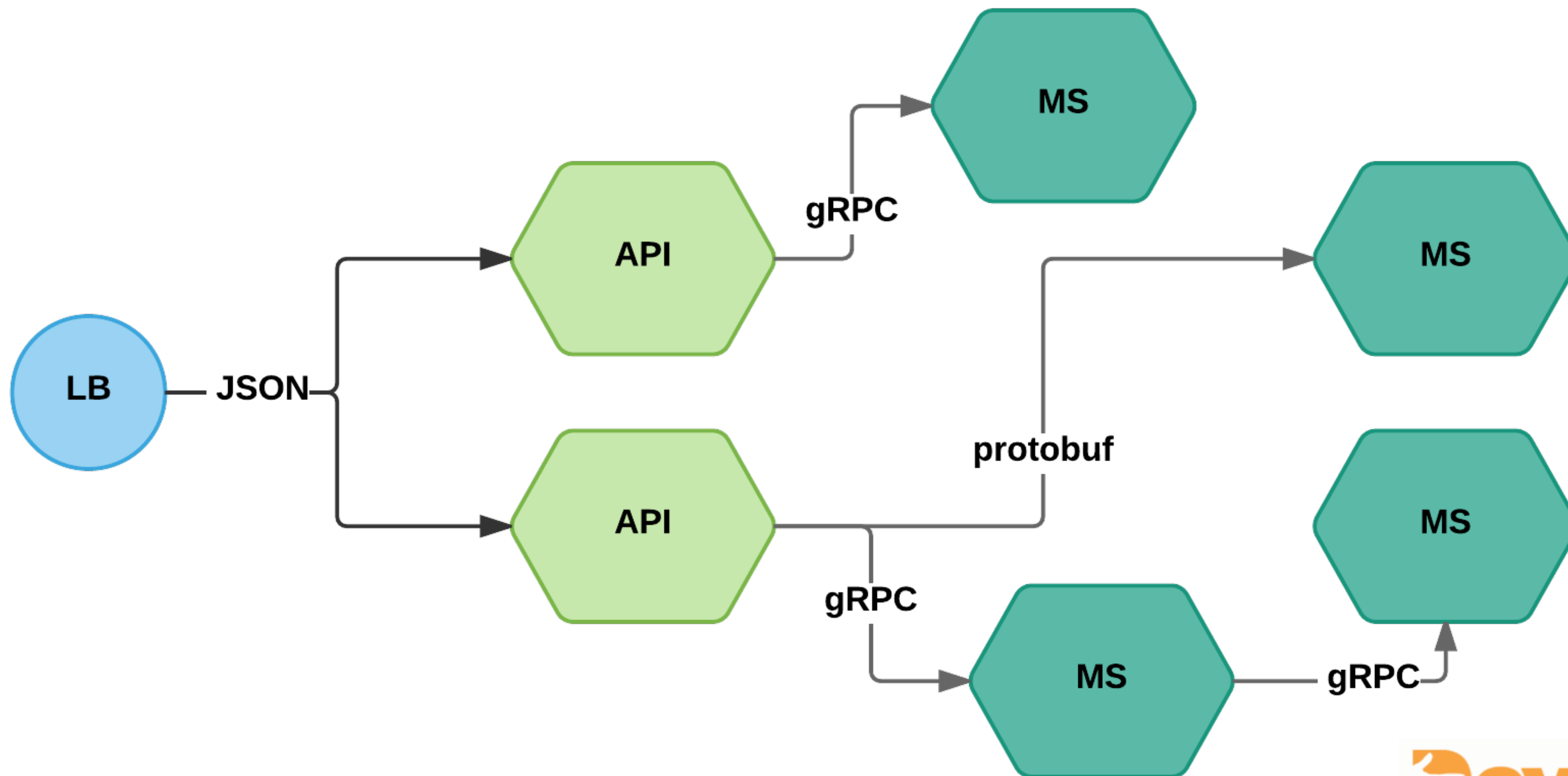
Журналирование

- ✓ Журналирование (12-факторное приложение)
<http://12factor.net/ru/logs>
- ✗ Logrus
<https://github.com/Sirupsen/logrus>
- ✓ xLog
<https://github.com/rs/xlog>
- ✓ Отдельно сразу в syslog хост-машины access и error логи

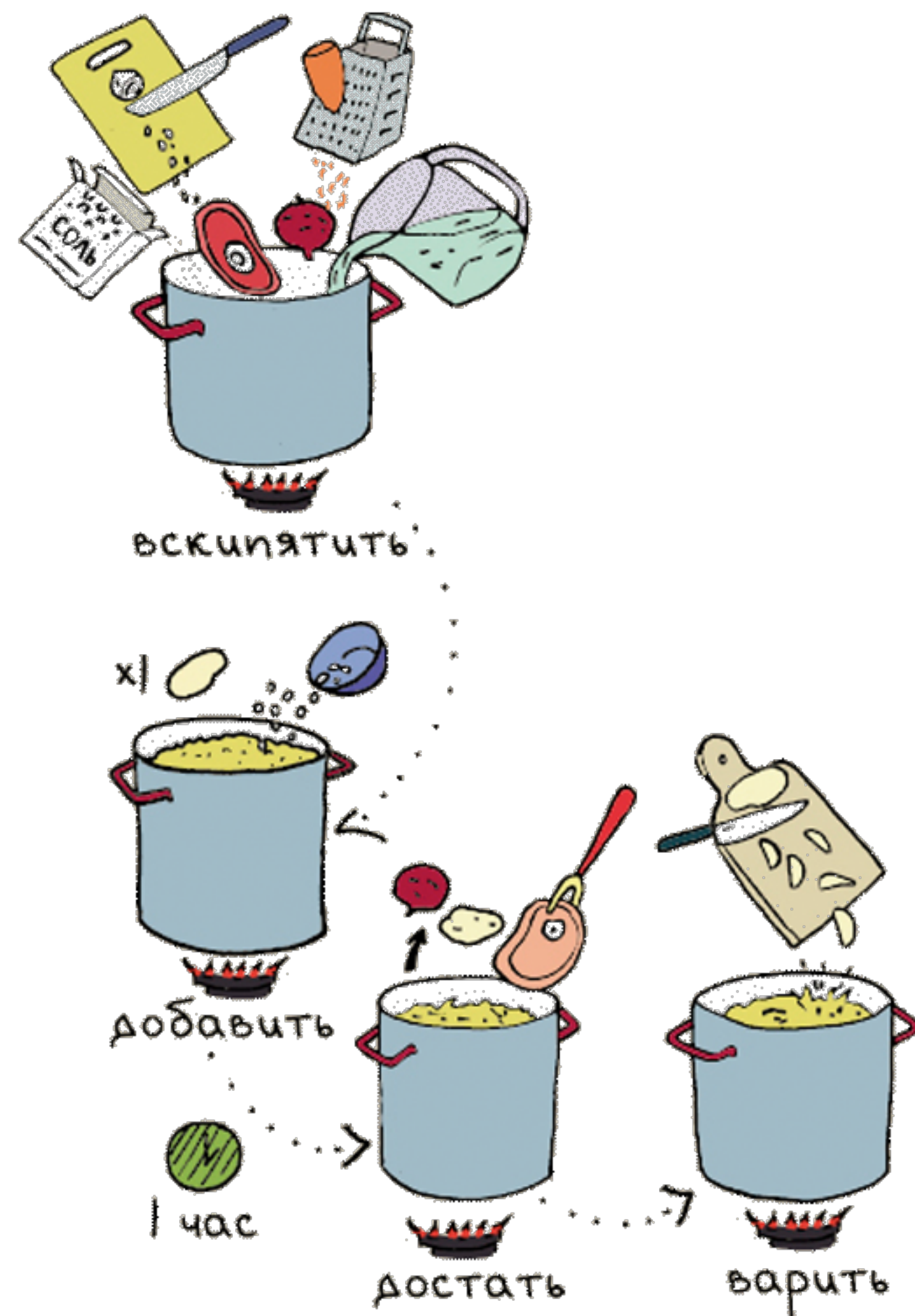
Транспорт



Транспорт



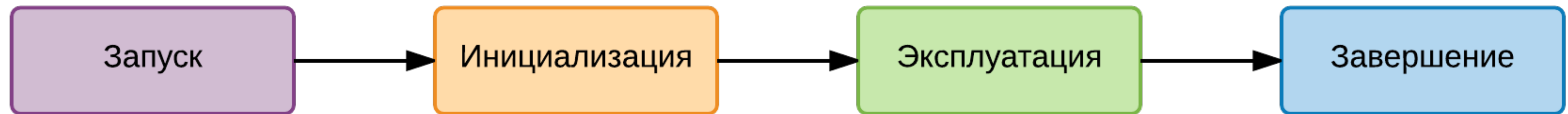
Health-check, метрики



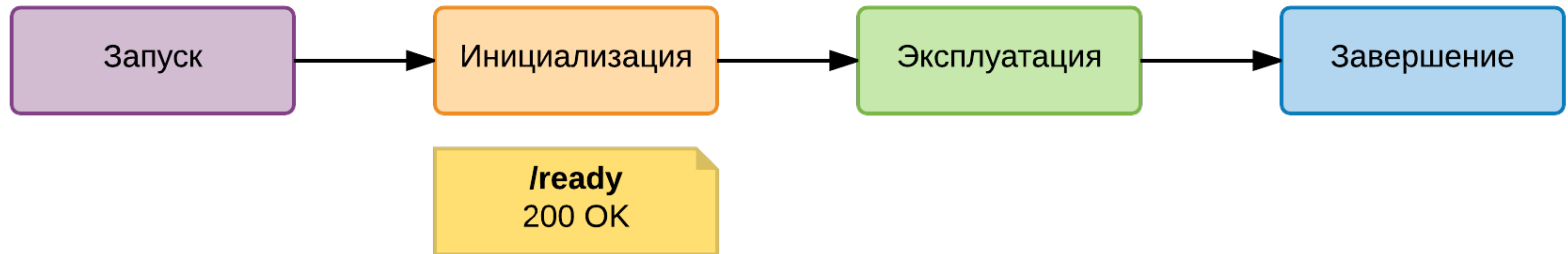
Health-check

- ✓ HTTP / HTTPS
- ✓ 2 основных метода
 - /ready
 - /health
- ✓ 1 опциональный
 - /info

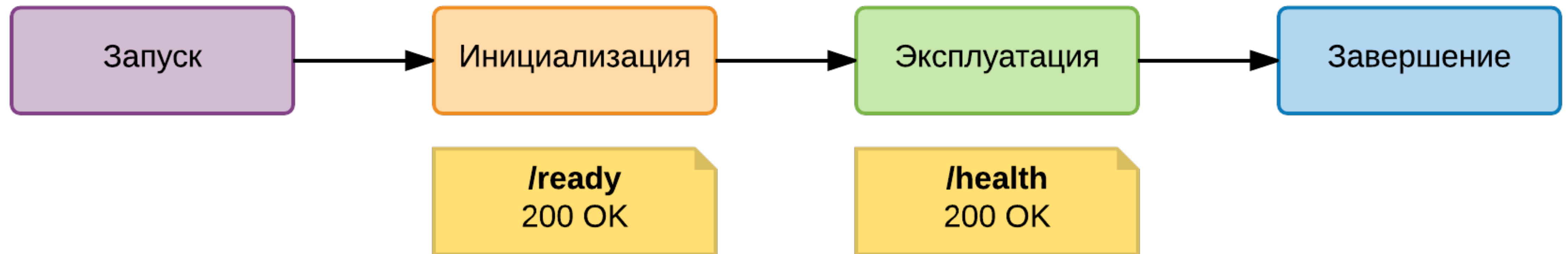
Health-check



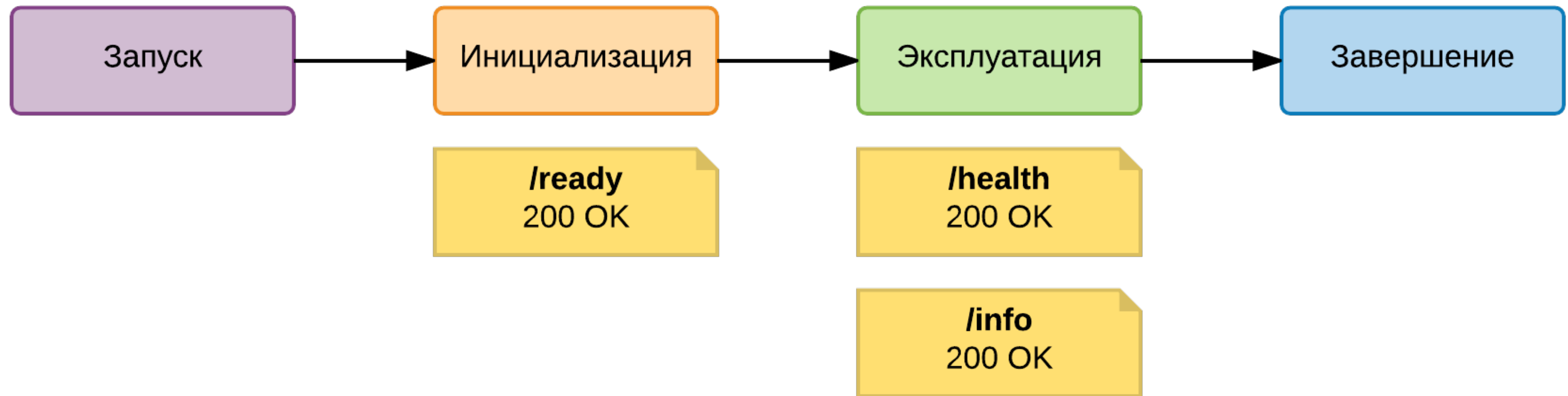
Health-check



Health-check



Health-check

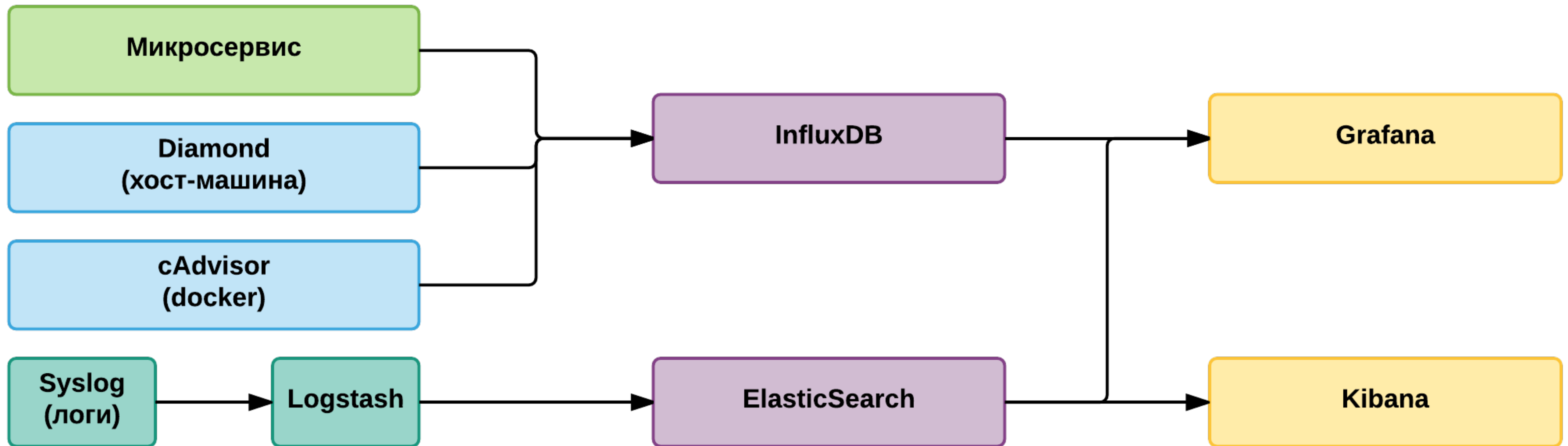


Health-check, сбор метрик

/info

```
{  
  "Name": "content-api",  
  "Version": "0.144",  
  "Uptime": 1778,  
  "Requests": {  
    "2XX": 12001,  
    "3XX": 1889,  
    "4XX": 122,  
    "5XX": 0,  
  }  
  "Status": 2,  
  "Payload": { ... }  
}
```

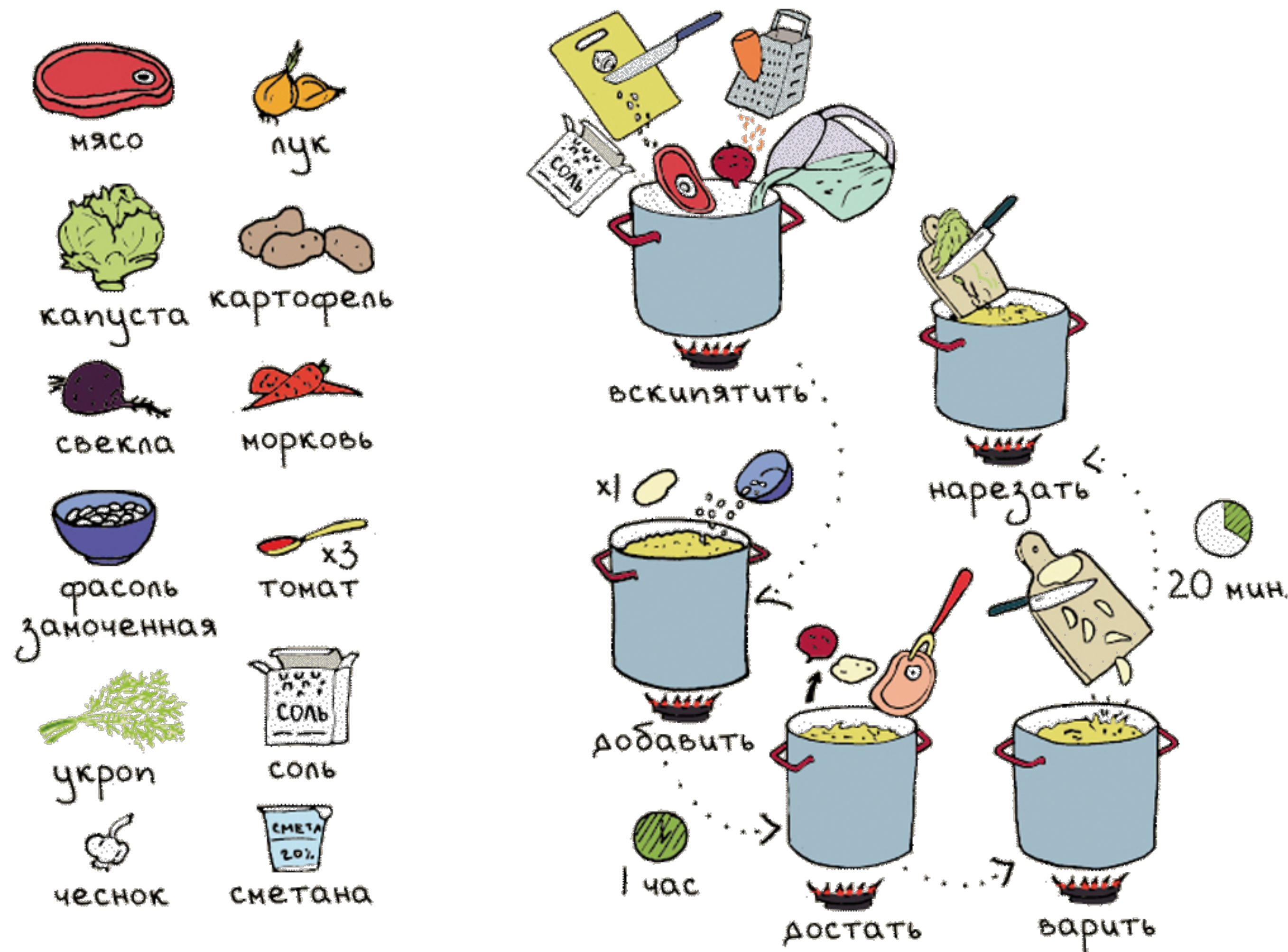
Сбор метрик



Сбор метрик

- ✓ Сбор метрик
<https://github.com/rcrowley/go-metrics>
Раз в 10 сек

Взаимодействие с сервисами



Взаимодействие с сервисами

Сервис назначения доступен, но:

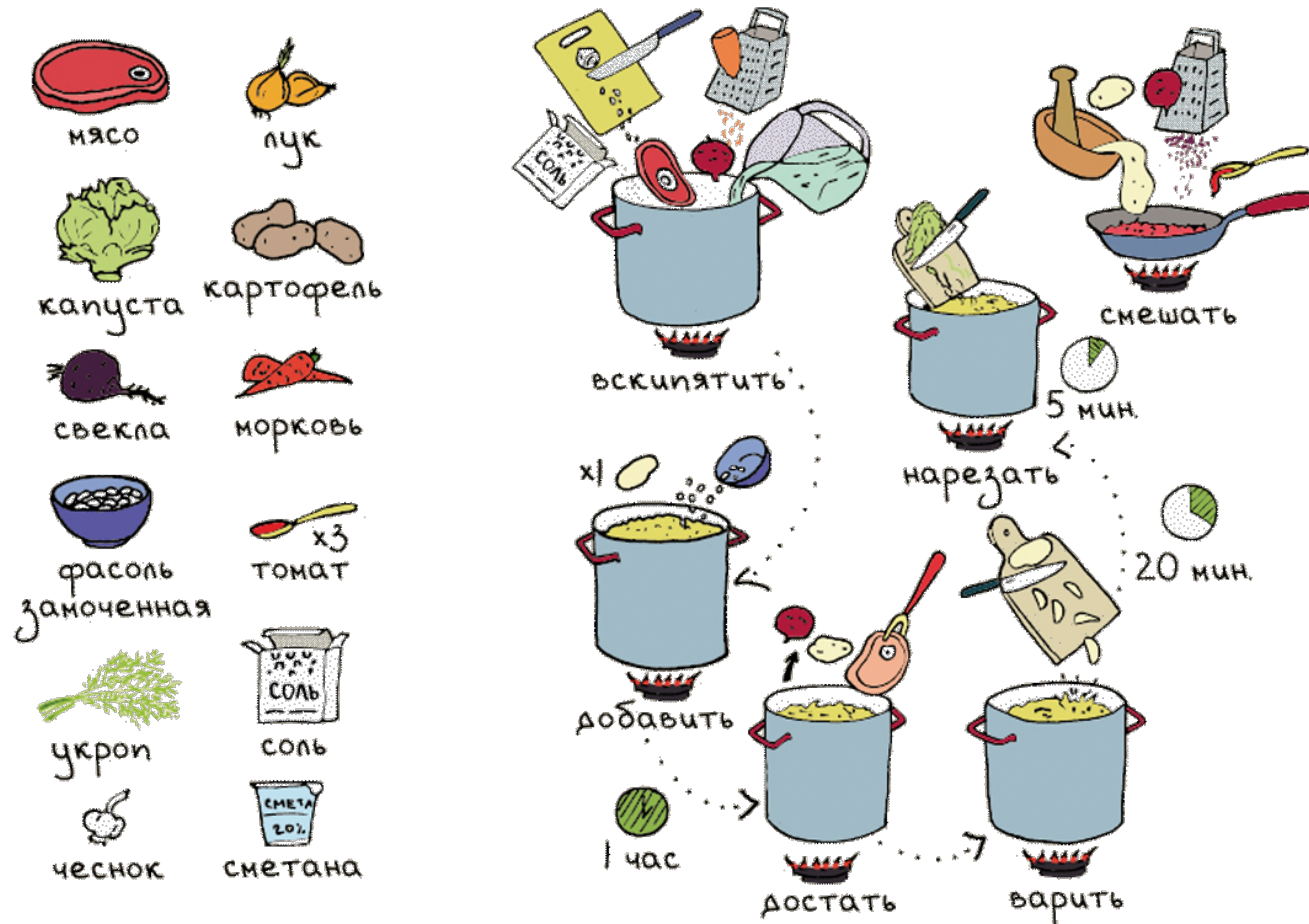
- Отвечает очень медленно
- Отвечает эпизодически
- Отвечает некорректно

Взаимодействие с сервисами

Паттерн Circuit Breaker (рубильник? прерыватель?)

- ✓ Централизованная обработка:
<https://github.com/Netflix/Hystrix>
- ✓ Клиент для Go:
<https://github.com/afex/hystrix-go>

Защита от перегруза



Защита от перегруза

- Входящими запросами
- Исходящими запросами

Защита от перегруза

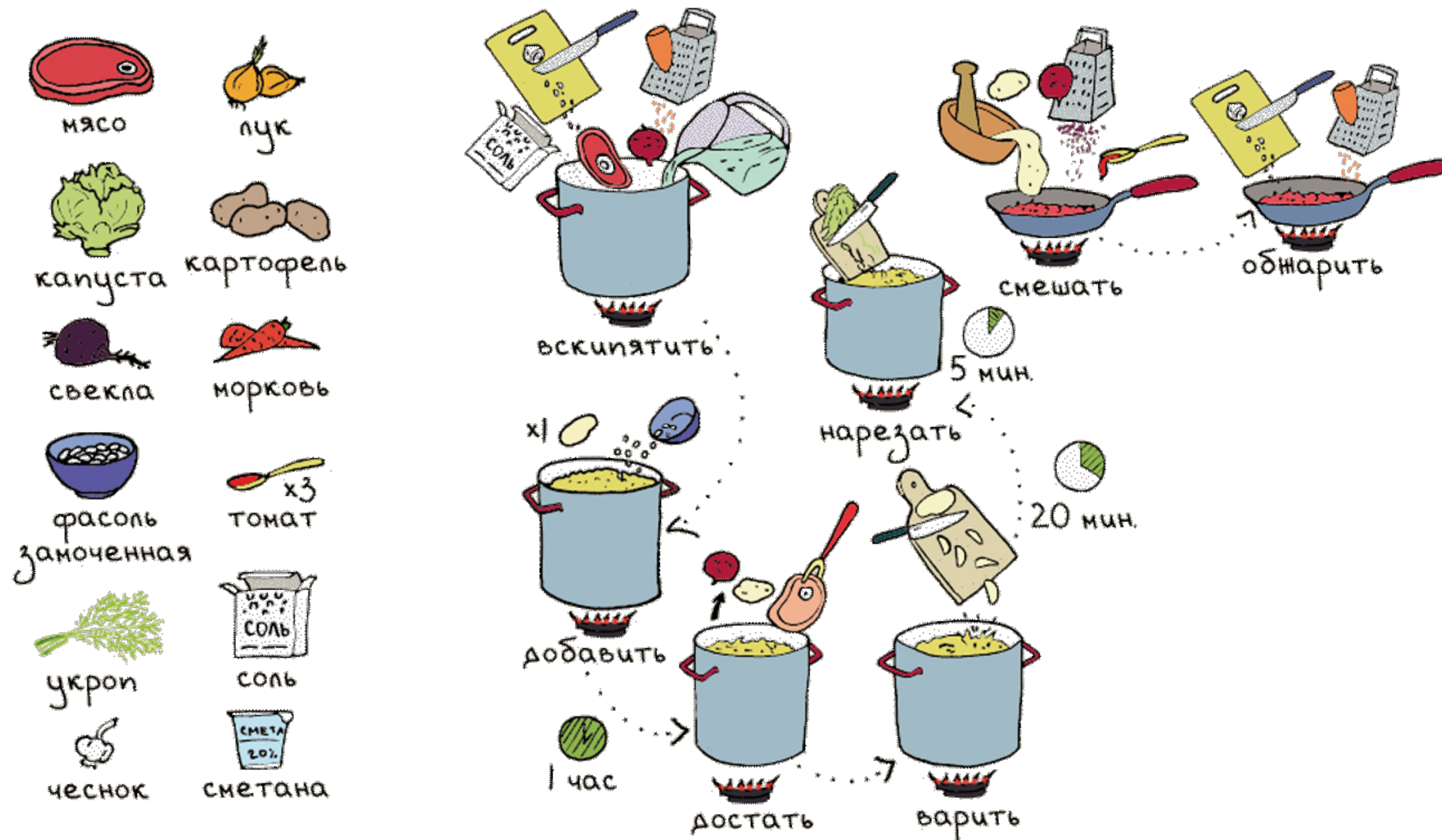
Общая механика:

https://ru.wikipedia.org/wiki/Алгоритм_текущего_ведра

Реализации:

- ✓ Golang Wiki:
<https://github.com/golang/go/wiki/RateLimiting>
- ✓ <https://github.com/juju/ratelimit>

Трассировка запроса



Duration: 195.000ms

Services: 8

Depth: 2

Total Spans: 17

Expand All

Collapse All

Filter Service Search

cms-lookup-server x1

oppdragserver x1

prod x1

search-management x3

smajobber-thrift-server x1

smajobberlookupservice x1

statistics x1

sybase: finnst x9



Трассировка запроса

Общая спецификация

<http://opentracing.io/>

Трассировка запроса

Общая спецификация

<http://opentracing.io/>

✗ AppDash

<https://github.com/sourcegraph/appdash>

<https://github.com/sourcegraph/appdash/tree/master/opentracing>

Трассировка запроса

Общая спецификация

<http://opentracing.io/>

✗ AppDash

<https://github.com/sourcegraph/appdash>

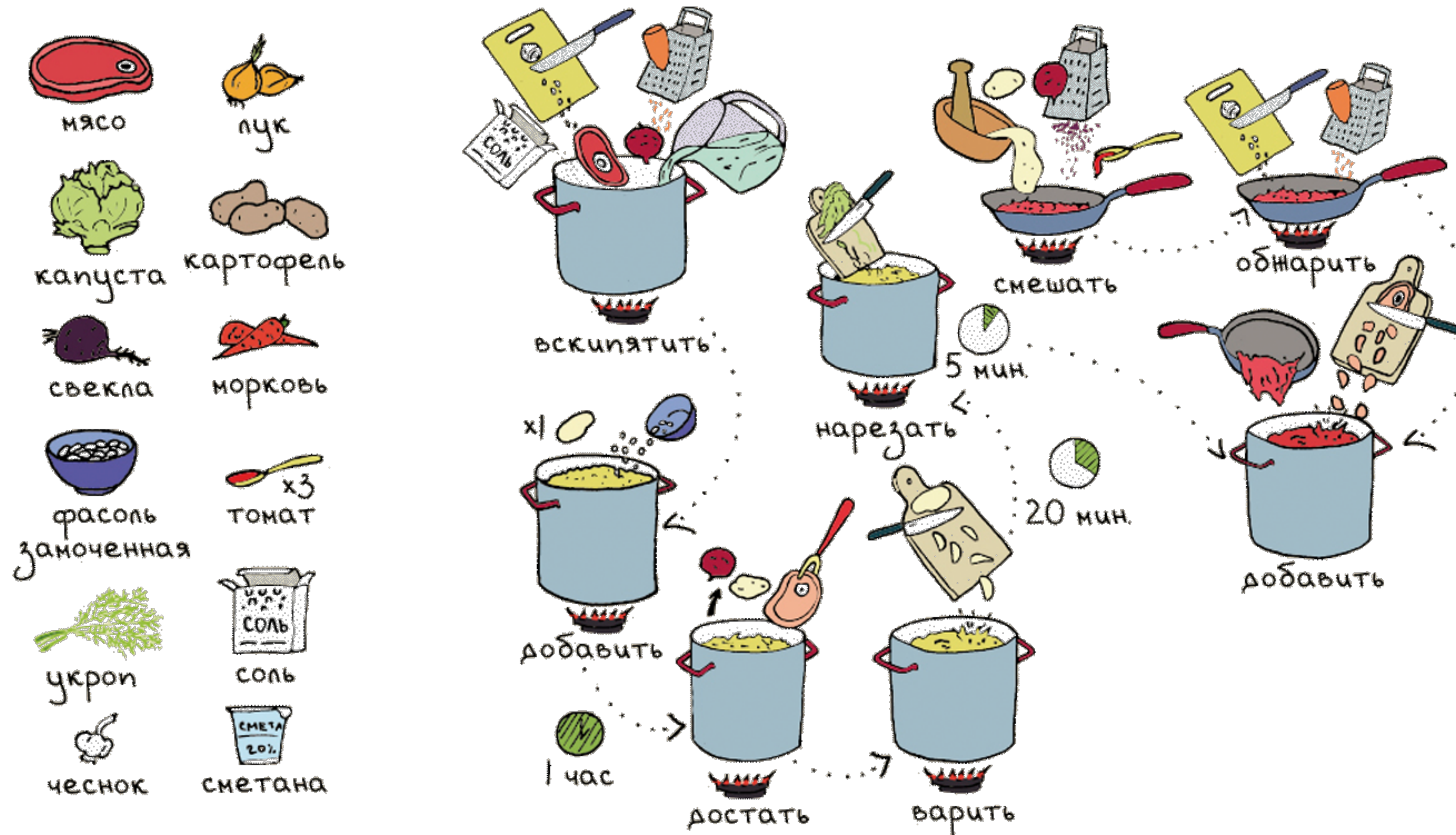
<https://github.com/sourcegraph/appdash/tree/master/opentracing>

✓ Zipkin

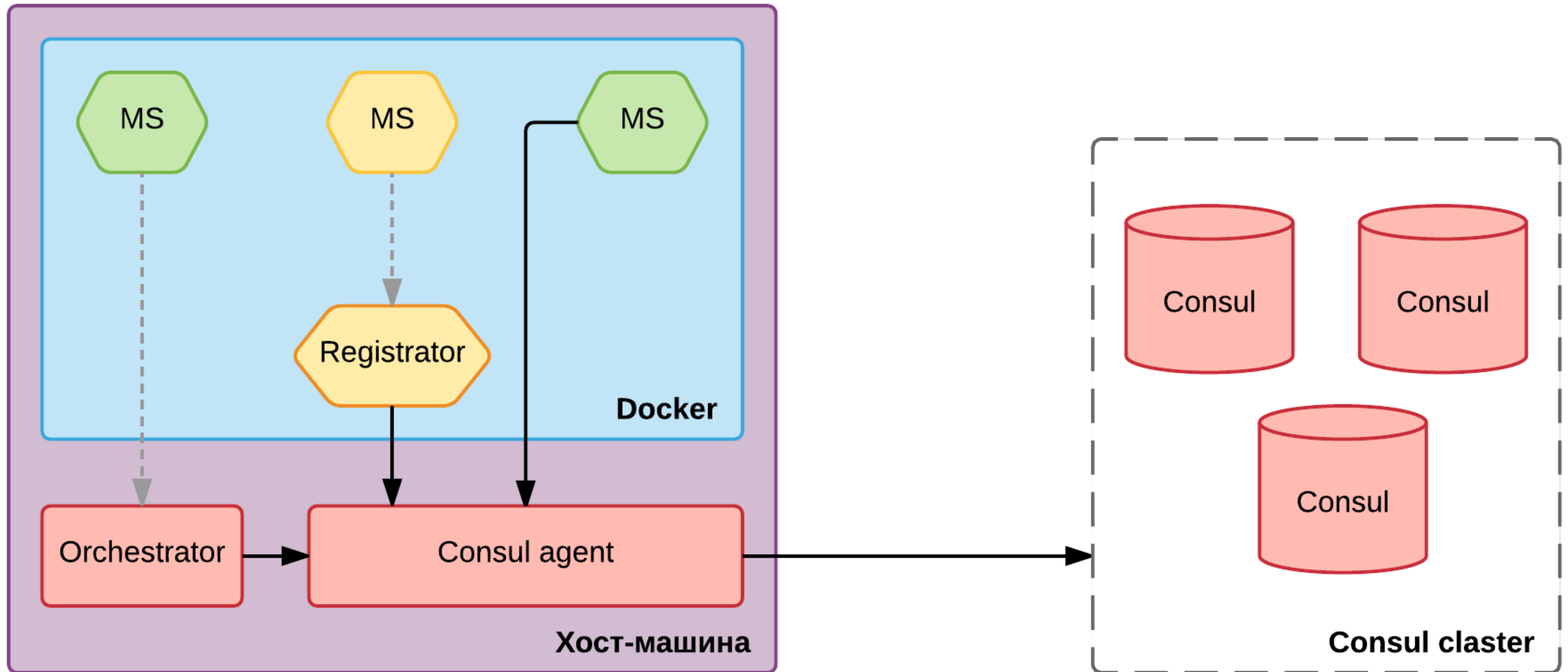
<https://github.com/openzipkin/zipkin>

<https://github.com/openzipkin/zipkin-go-opentracing>

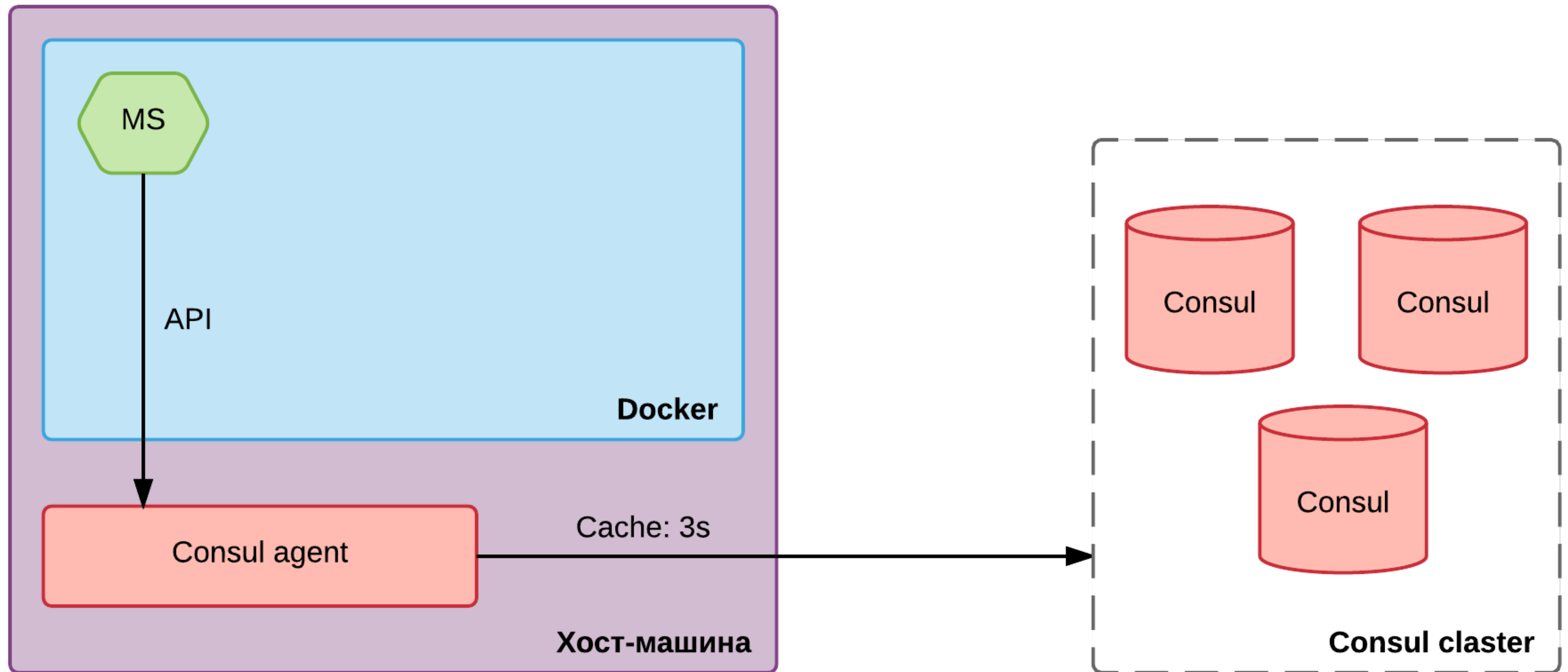
Обнаружение сервисов



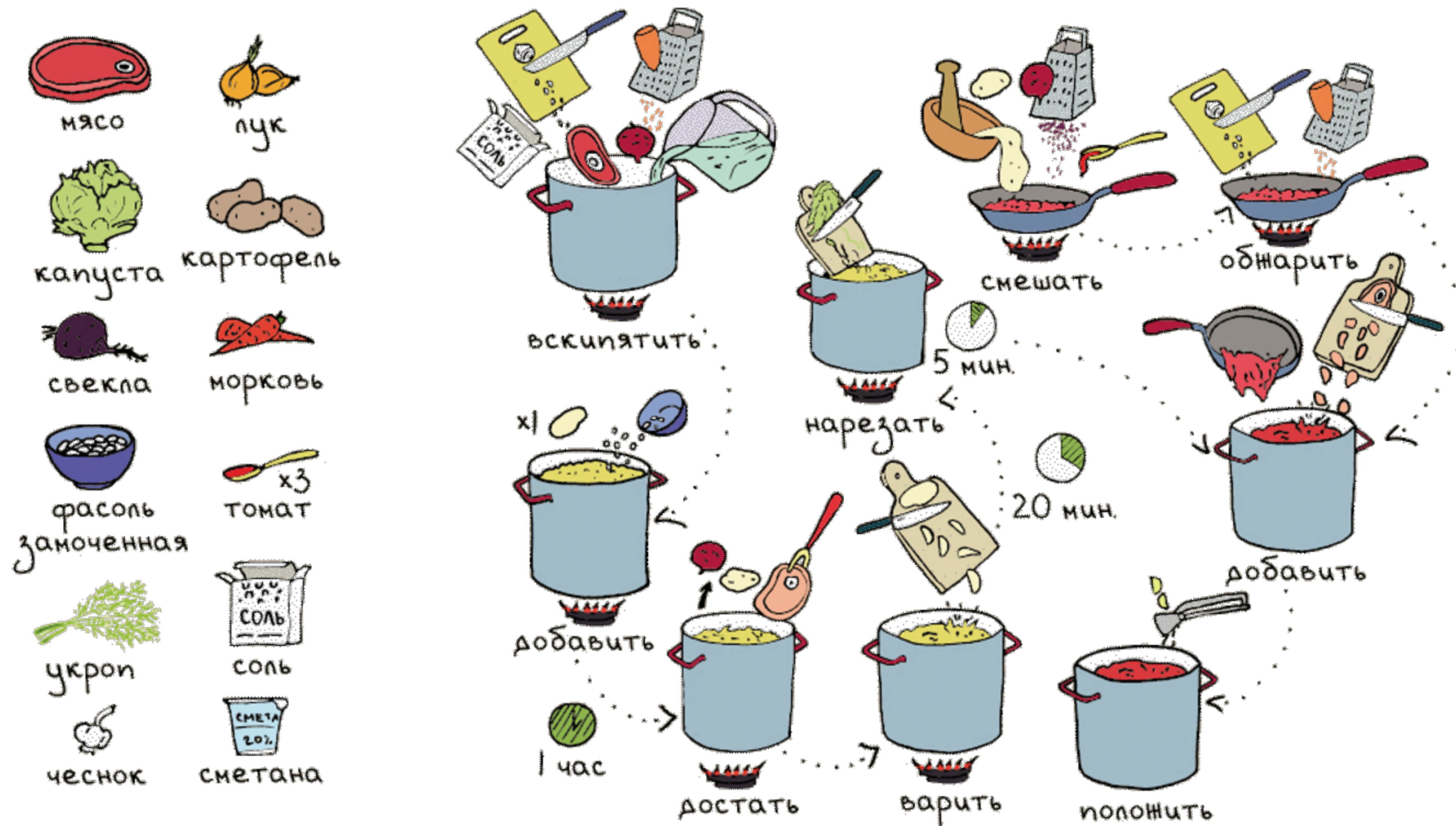
Обнаружение сервисов



Обнаружение сервисов



Масштабирование



В общем случае — миф!

Масштабирование

Микросервисы

- не имеют состояния
- не привязаны к машине, на которой запускаются

Масштабирование

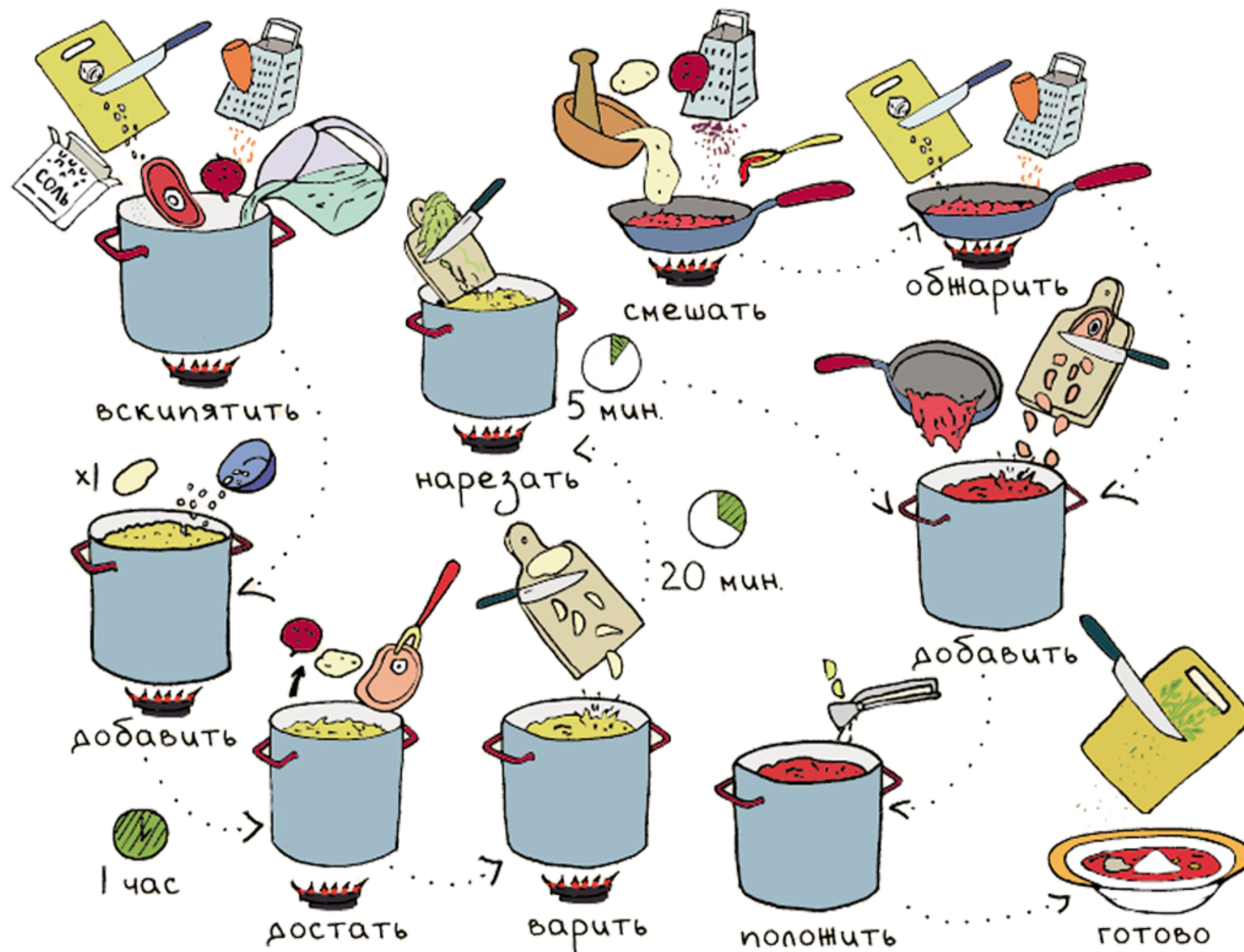
Микросервисы

- не имеют состояния
- не привязаны к машине, на которой запускаются

Паттерны масштабирования данных

- master—slave
- шардирование

Борщ ... эээ ... Сервис готов!



Спасибо!

Вадим Мадисон

M-Tex

vmadison@rutube.ru