

Аналитическое тестирование



Тестируем API с использованием динамических данных

О чем поговорим

- Немного о тестировании
- Методы тестирования со статическими данными
- Тестирование с динамическими данными:
 - О чем же это?
 - Примеры



Суровцев Иван

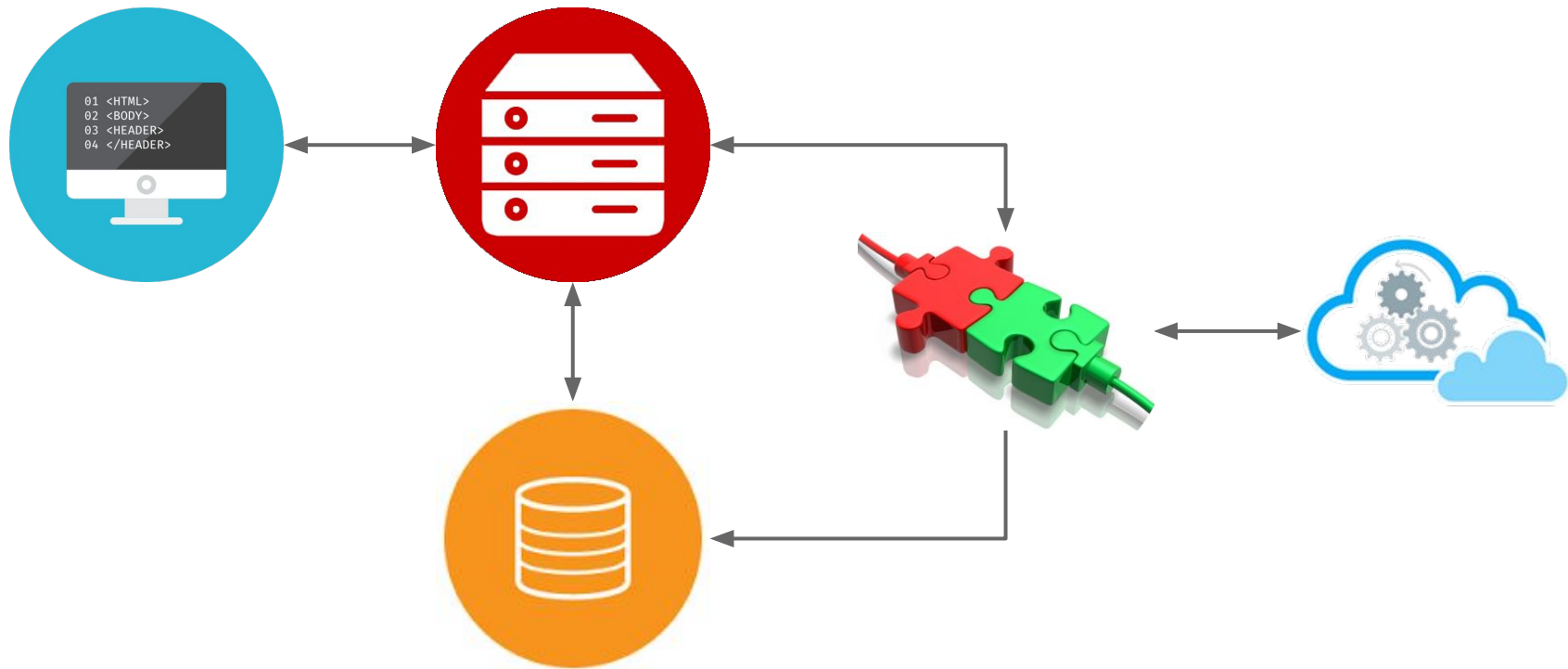
i.surovtsev@semrush.com

Инженер по тестированию и
автоматизации тестирования

Position Tracking Tool



Сервис



Этапы тестирования

- Получаем данные для тестов
- Анализируем данные
- Составляем наборы данных
- Проводим тесты
- Обрабатываем полученные результаты

Как обычно автоматизируем

- Составляем наборы данных
- Проводим тесты
- Обрабатываем полученные результаты

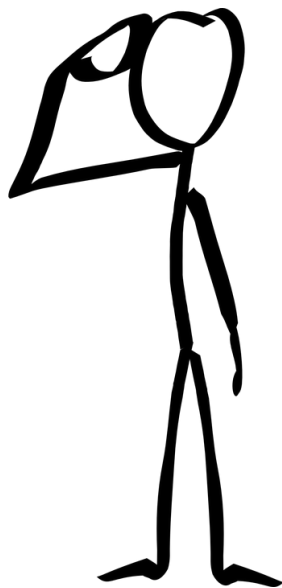
Стандартные методы тестирования

Тестирование с данными загружаемыми в БД:

- + Независимость тестов от конфигурации окружения
- + Стабильность выполнения
- + Скорость
- Нужен полный доступ к БД
- Чистота окружения
- Релевантность данных

Стандартные методы тестирования

А если нет доступа к БД?



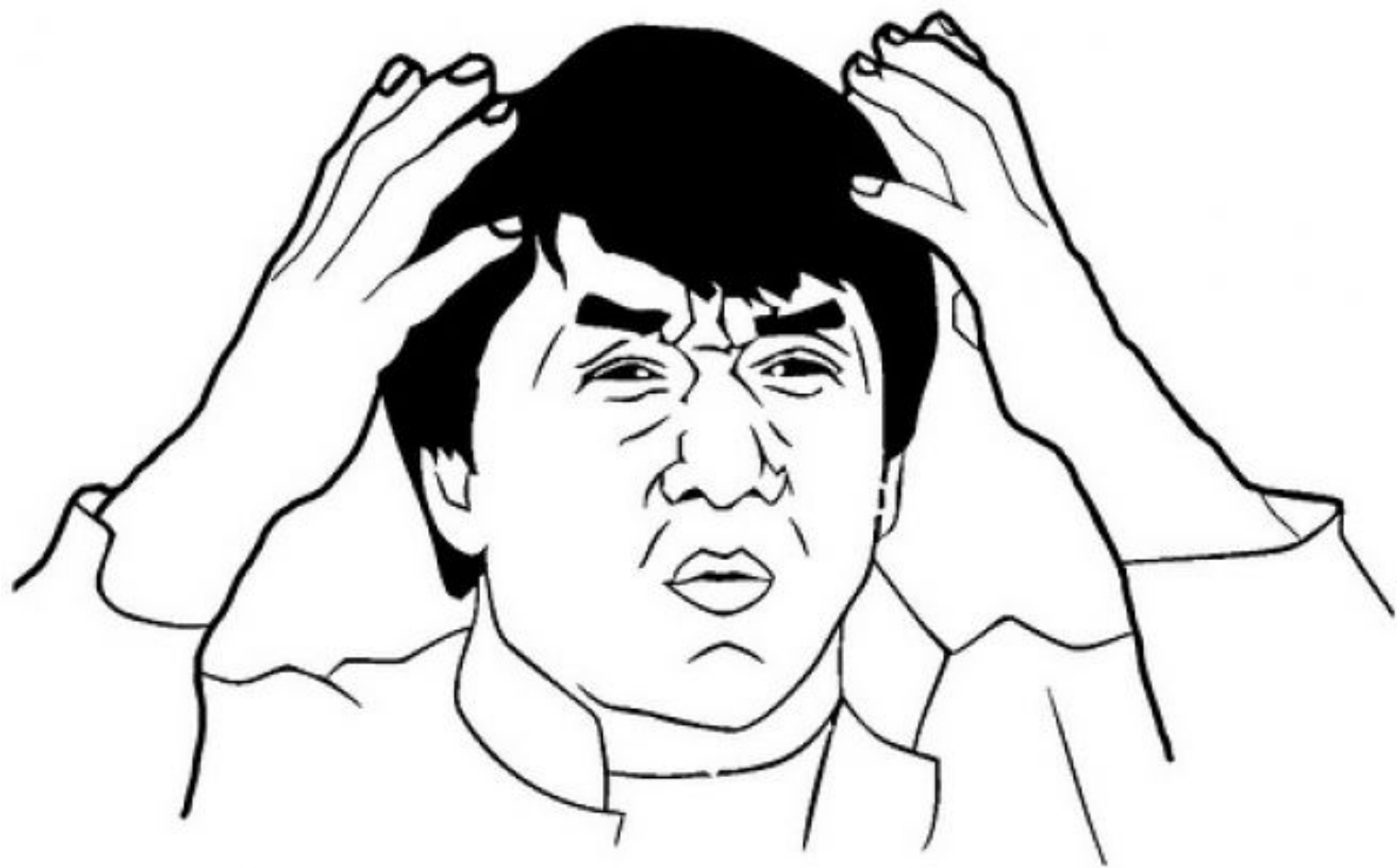
Стандартные методы тестирования

Готовим данные через интерфейсы:

- Необходимо проверять процесс подготовки данных
- Не можем гарантировать проверку всеми тестами

Внешние источники данных

- Внешний сервис без доступа, только запросы
- Нет возможности изменить данные
- Данные меняются каждый день
- Состав данных меняется периодически



Стандартные методы тестирования

Заглушки:

То же что и с доступом к БД, а также

- + Имитация внешнего источника
- Отдельное приложение, которое необходимо обновлять и поддерживать

Аналитическое тестирование

Автоматизируем

- Получаем данные для тестов
- Анализируем
- Составляем наборы данных

Аналитическое тестирование

id:"492075", uri:"domain.com"

domain.com

id:"310865", uri:"apple.com"

com

id:"412906", uri:"therooster.lol"

id:"438678", uri:"http://apple.com"

apple

id:"519324", uri:"http://xn--d1acufc5f.xn--p1ai"

xn--d1acufc5f

Аналитическое тестирование

id:"492075", uri:"domain.com"

id:"310865", uri:"apple.com"

id:"412906", uri:"therooster.lol"

} 3

un:"Ivan", type: "pro", campaigns: 5, keywords: 500 ,...

Аналитическое тестирование

Что получаем:

- + Данные в БД проходят через анализ на “чистоту”
- + Актуальные и настоящие данные
- + Валидация наборов данных при помощи правил

Аналитическое тестирование

Недостатки:

- Нет возможности увидеть данные перед тестом
- Появляется желание применять во всех тестах
- ~ Тесты зависимы от источников данных

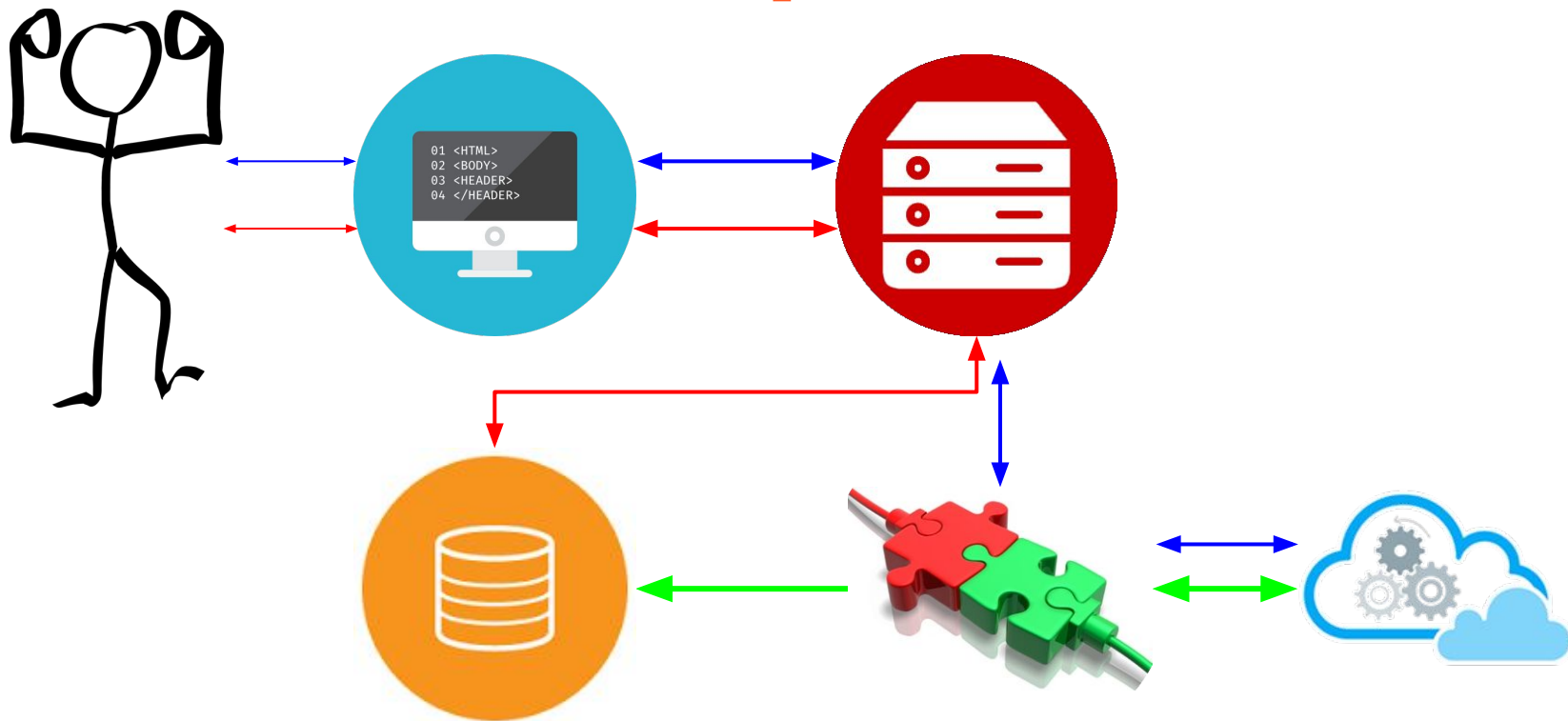
Немного банальных правил

- Максимальная простота
- Не анализировать все подряд, это не всегда уместно

Источники данных

- Ввод пользователя
- Внешние сервисы
- Внутренние справочники
- Результат работы ПО

Движение данных в сервисе



Источники данных в сервисе

- Домен - используем эталон
- Локация - выбираем из справочника
- Ключевые слова - рандом основанный на пункте 1

Запускаем кампанию

Тестовые данные

Локация: Saint Petersburg, Saint Petersburg, Russia

Домен: myawesomeplace.com

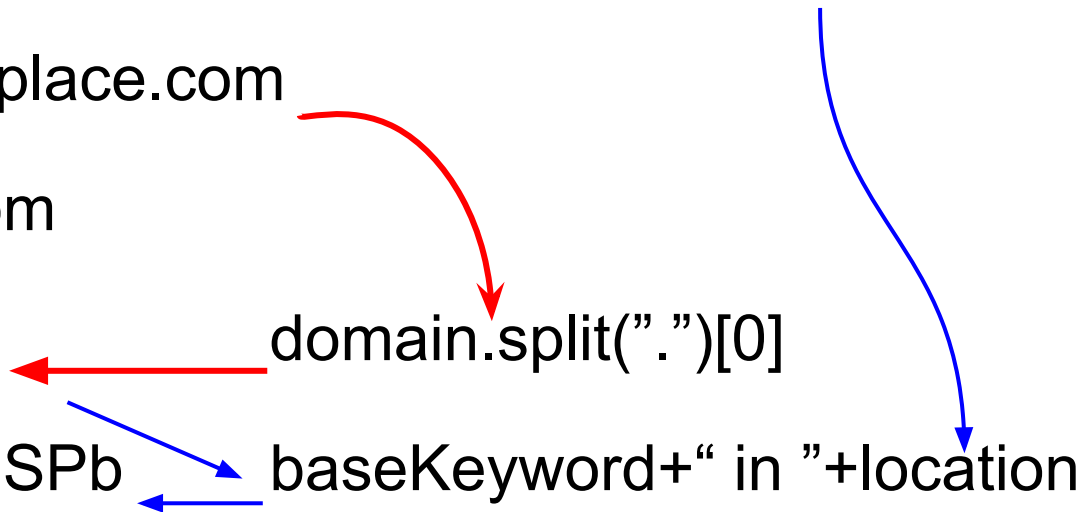
myawesomeplace.com

myawesomeplace

myawesomeplace in SPb

`domain.split(".")[0]`

`baseKeyword+" in "+location`

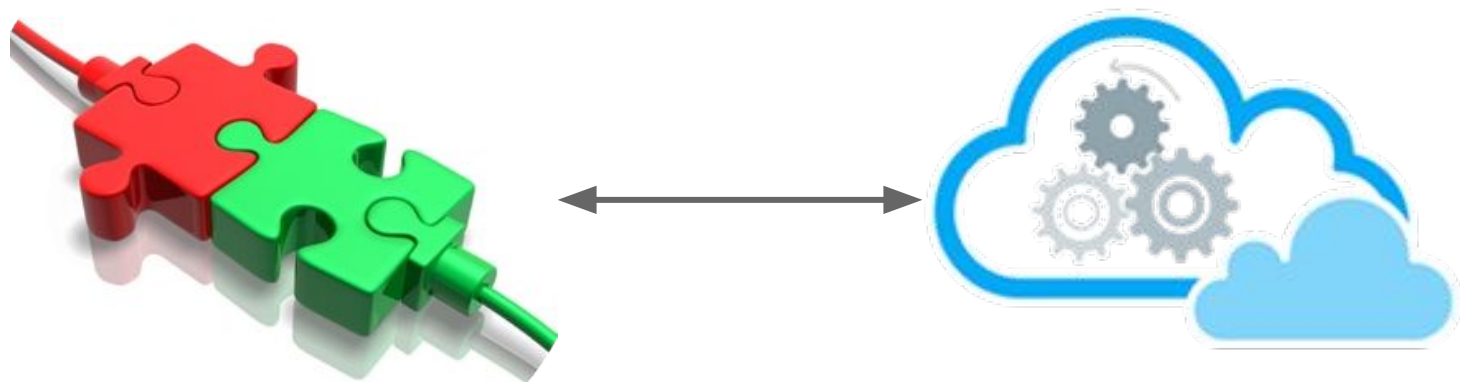


Источники данных в сервисе

После запуска кампании

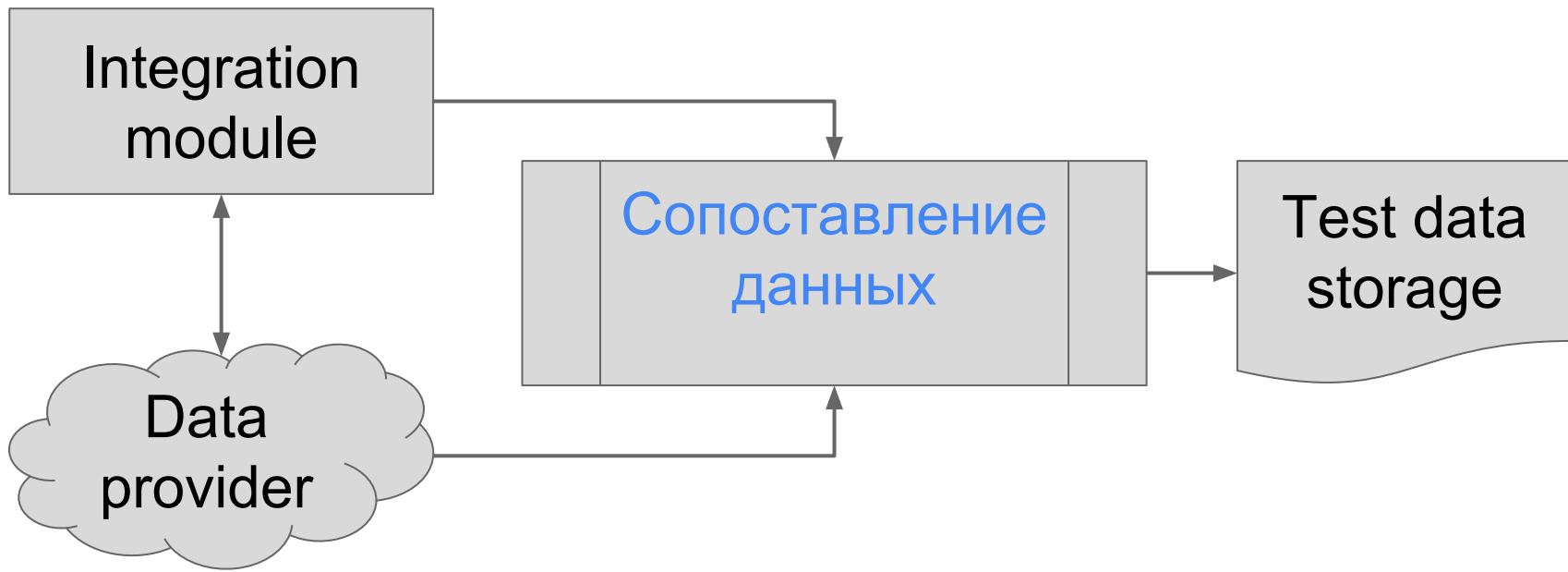
- Поисковые страницы - внешние источники
- Позиции доменов - внешние источники
- Метрики - результат работы сервиса
- Исторические данные - результат работы сервиса

Тестирование **Integration Module**



Тестирование **Integration Module**

Пример 1. Сопоставление данных



Тестирование **Integration Module**

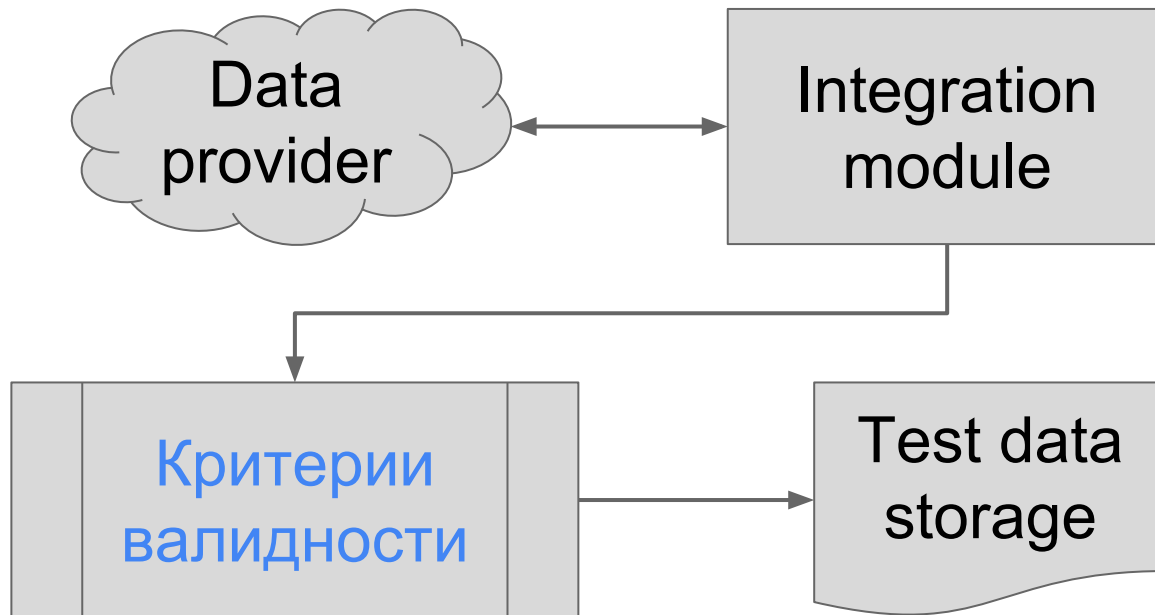
Пример 1. Сопоставление данных

```
DP: [
  {
    "uri":"http://competitor.org/index.htm",
    ...
  },
  {
    "uri":"https://myawesomeplace.com",
    ...
  },
  ...
]

IM: [
  {
    "competitor.org":"1",
    ...
  },
  {
    "myawesomeplace.com":"2",
    ...
  },
  ...
]
```

Тестирование **Integration Module**

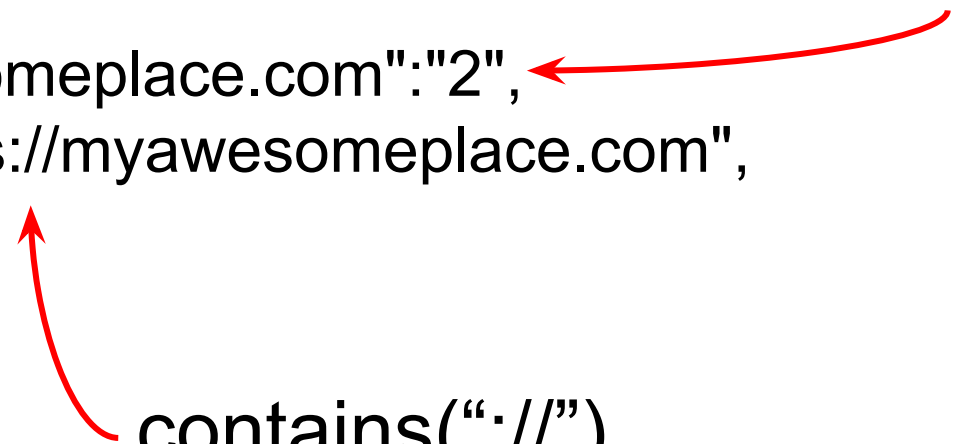
Пример 2. Критерии валидности



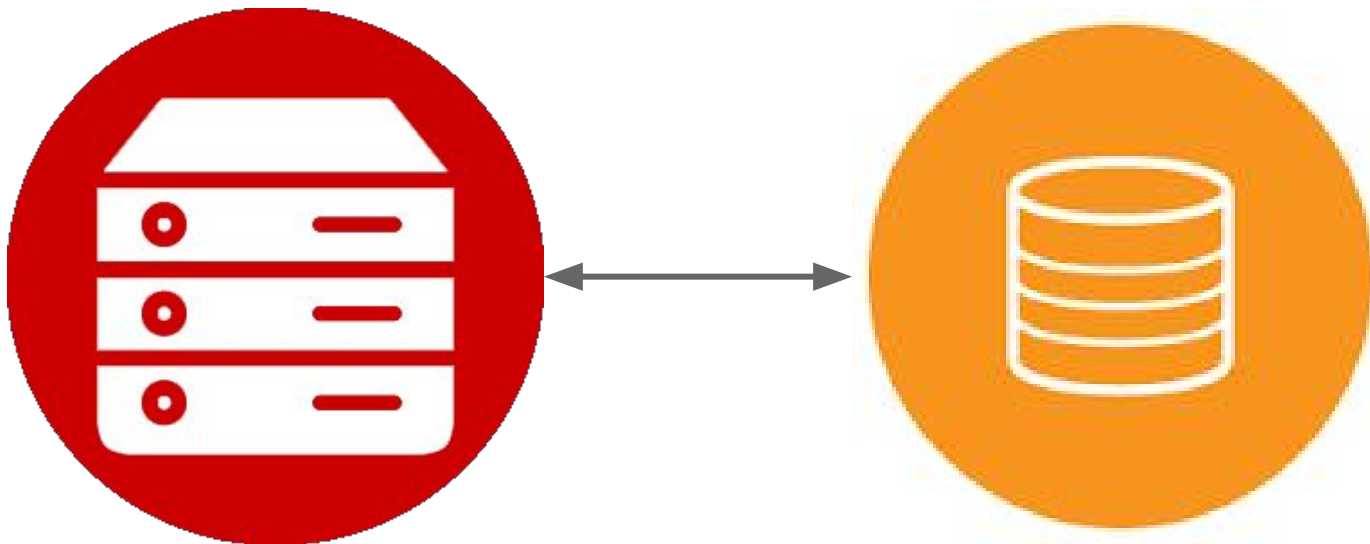
Тестирование **Integration Module**

Пример 2. Критерии валидности

```
IM: [ ...,           != null && != 0 && != ""  
  {  
    "myawesomeplace.com":"2",  
    "uri":"https://myawesomeplace.com",  
    ...  
  }, ...  
]
```

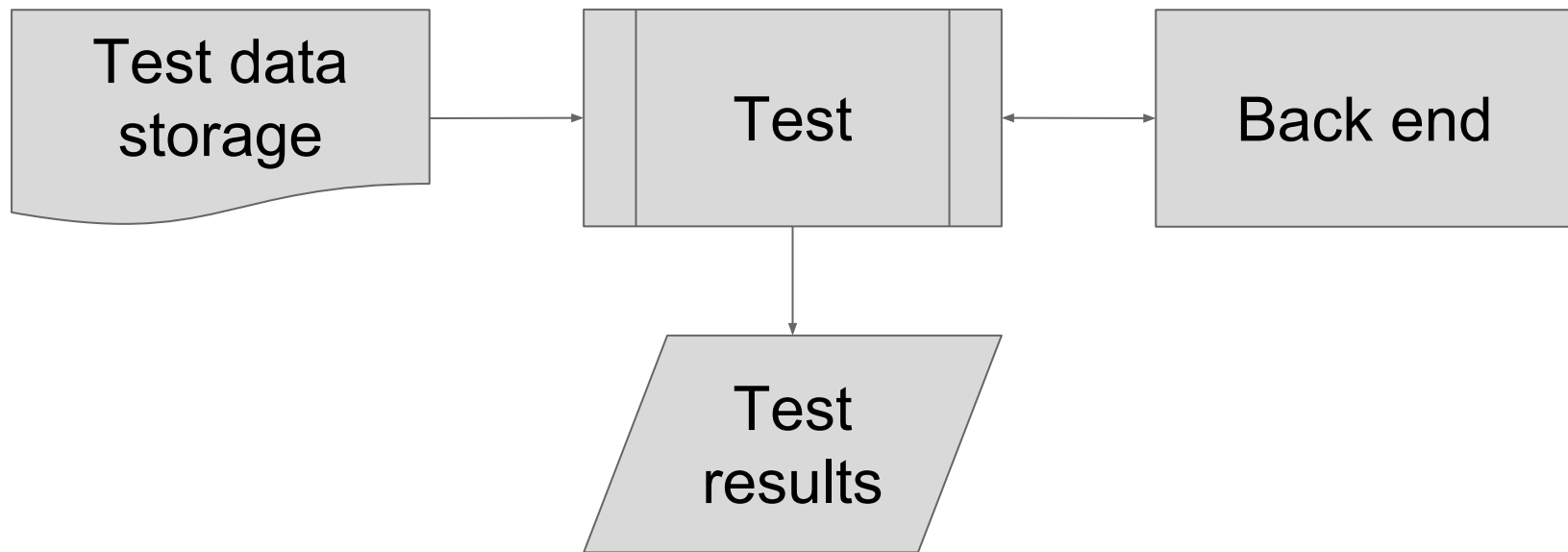
 contains("://")

Тестирование **back end**



Тестирование **back end**

Пример 1. Используем ранее полученные данные



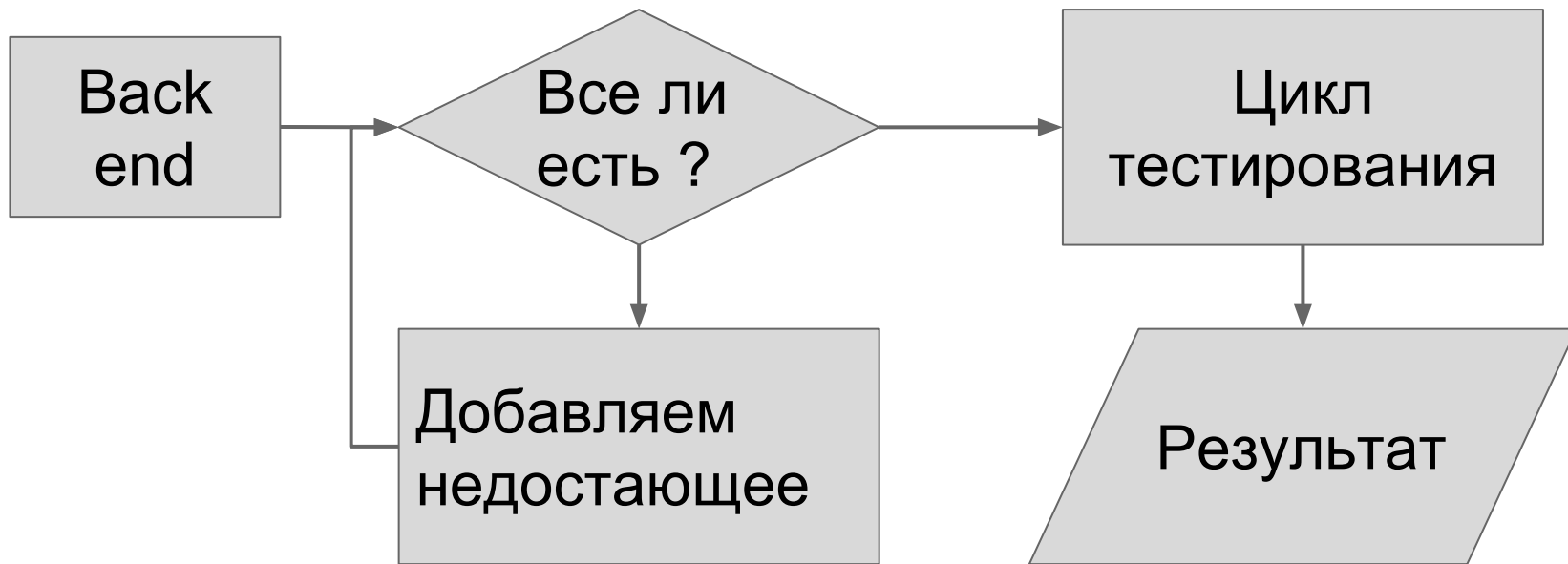
Тестирование **back end**

```
Storage: [ ..., {  
  "myawesomeplace.com": "2",  
  ...  
}, ...]
```

```
Back end: {  
  "domain": "myawesomeplace.com",  
  "data": {  
    "20161203": 3,  
    "20161204": 2  
  }  
}
```

Тестирование **back end**

Пример 2. Проверяем полноту данных



Тестирование **back end**

Пример 2. Проверяем полноту данных

```
[ {  
  "domain":"myawesomeplace.com",  
  "region":"Saint Petersburg"  
},  
{  
  "domain":"apple.com",  
  "region":"Moscow"  
}]
```

Тестирование **back end**

Пример 3. Проверка граничных значений:

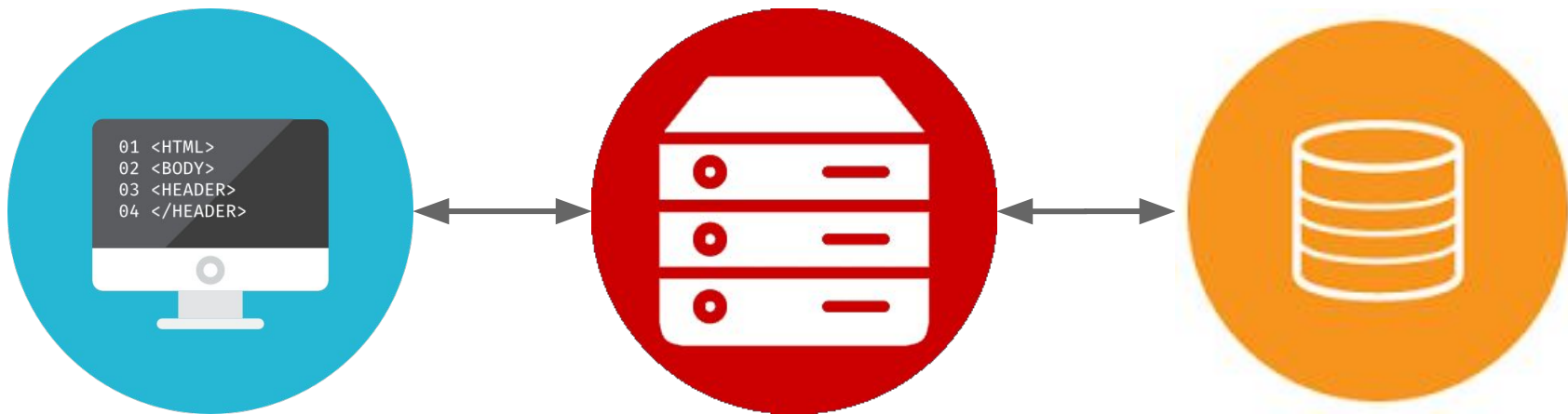
- Запрос лимитов
- Запрос количества существующих сущностей
- Определяем правила граничных значений

Тестирование **back end**

Как использовать для нагрузки:

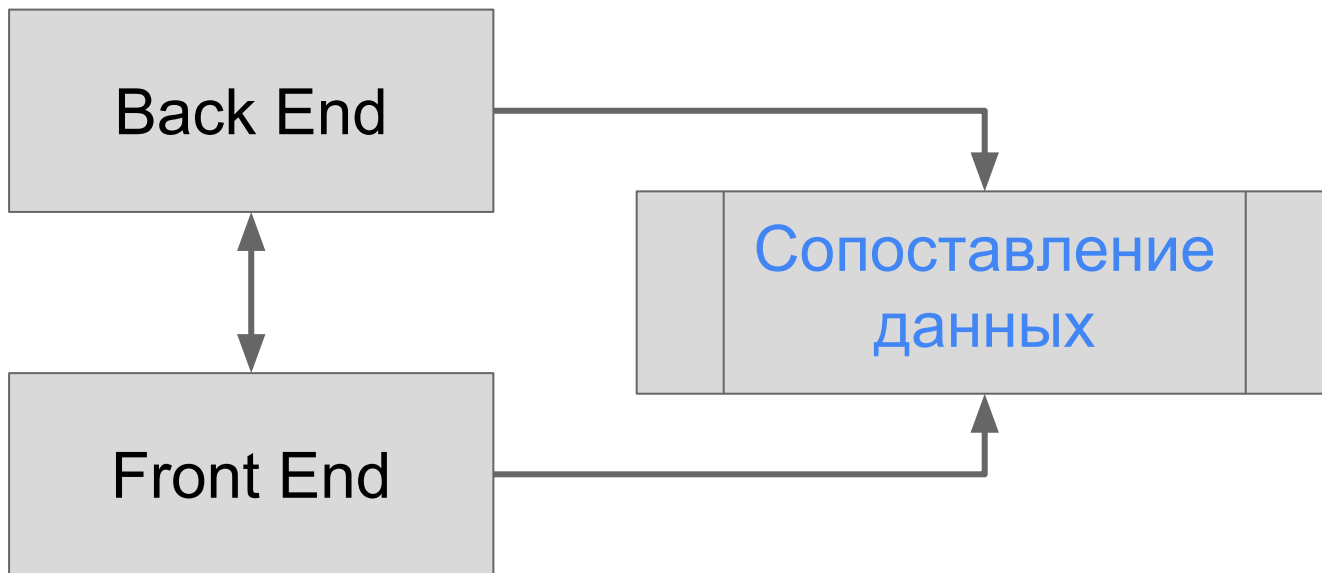
- Разделить данные на наборы
- Модифицировать и размножить

Тестирование **UI**



Тестирование UI

Пример 1. Сопоставление



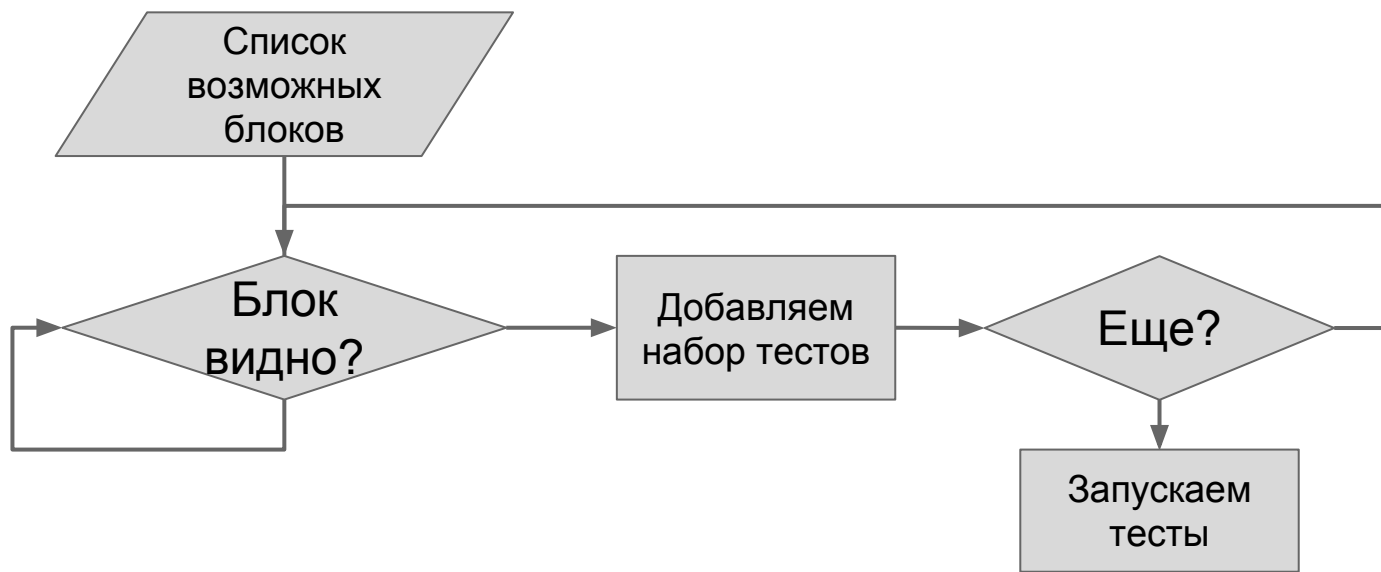
Тестирование **UI**

Пример 2. Фильтрации

- Получаем данные
- Выбираем уникальные значения
- Используем в тесте

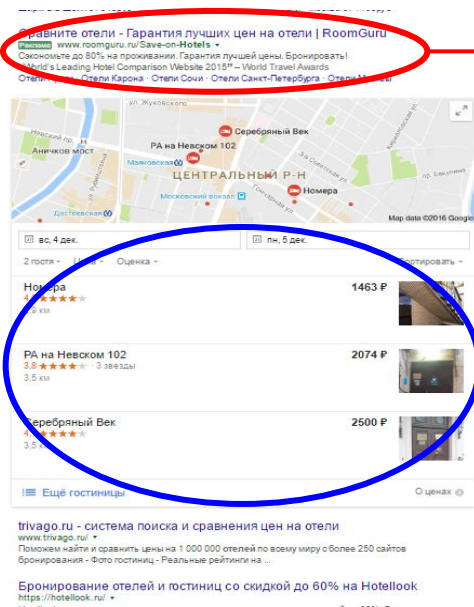
Тестирование UI

Пример 3. Тестируем отображаемый контент



Тестирование UI

Пример 3. Тестируем то, что видим



```
add(new XmlClass(  
    "com.tracking.serp.TestAdWords"))
```

```
add(new XmlClass(  
    "com.tracking.serp.TestLocal"))
```


Тестирование **UI**

Пример 4. Использование промежуточных данных

- Открываем список
- Запоминаем отображаемые значения
- Открываем детализацию
- Сопоставляем с ранее полученными

Итоги

Чтобы использовать надо определить:

- Источники данных приложения
- Каким типом можно заменить в тестах
- Есть ли связь между данными
- Как данные используются в приложении

Спасибо за внимание