

## Взаимодействие программ по сети

Чтобы предоставить программе доступ к информации или к возможностям другой программы, используют API, программные интерфейсы приложений. Зачастую программы и сервисы находятся в разных адресных пространствах: они удалены друг от друга, размещены на разных компьютерах или серверах. Взаимодействие таких программ основано на удалённом вызове процедур или функций: одна система отправляет запрос к другой, и в результате в удалённой системе вызывается функция.

Существует целый класс технологий, позволяющих программам удалённо вызывать функции или процедуры. Эти технологии называют RPC (англ. Remote Procedure Call, «удалённый вызов процедур»).

Рассмотрим самые известные технологии этого класса, которые чаще всего применяются для создания API.

### Протокол SOAP: данные в конверте

SOAP (англ. *Simple Object Access Protocol*, «простой протокол доступа к объектам») — это протокол, который применяется для удалённого вызова процедур, для обмена произвольными сообщениями в формате XML и для организации API-сервисов. Протокол SOAP был разработан для Microsoft, первый релиз появился в 1998 году.

Сообщения, которые курсируют между клиентом и сервером, — это XML-файлы, составленные по определённому стандарту:

- **Конверт** — это начальный и конечный теги сообщения.
- **Заголовок** — содержит необязательные атрибуты сообщения.
- **Тело** — содержит данные, которые сервер передаёт получателю.
- **Ошибка** — содержит информацию об ошибках, возникших при обработке сообщения.

Вот как может выглядеть XML, в котором по протоколу SOAP пересылается информация о двух студентах:

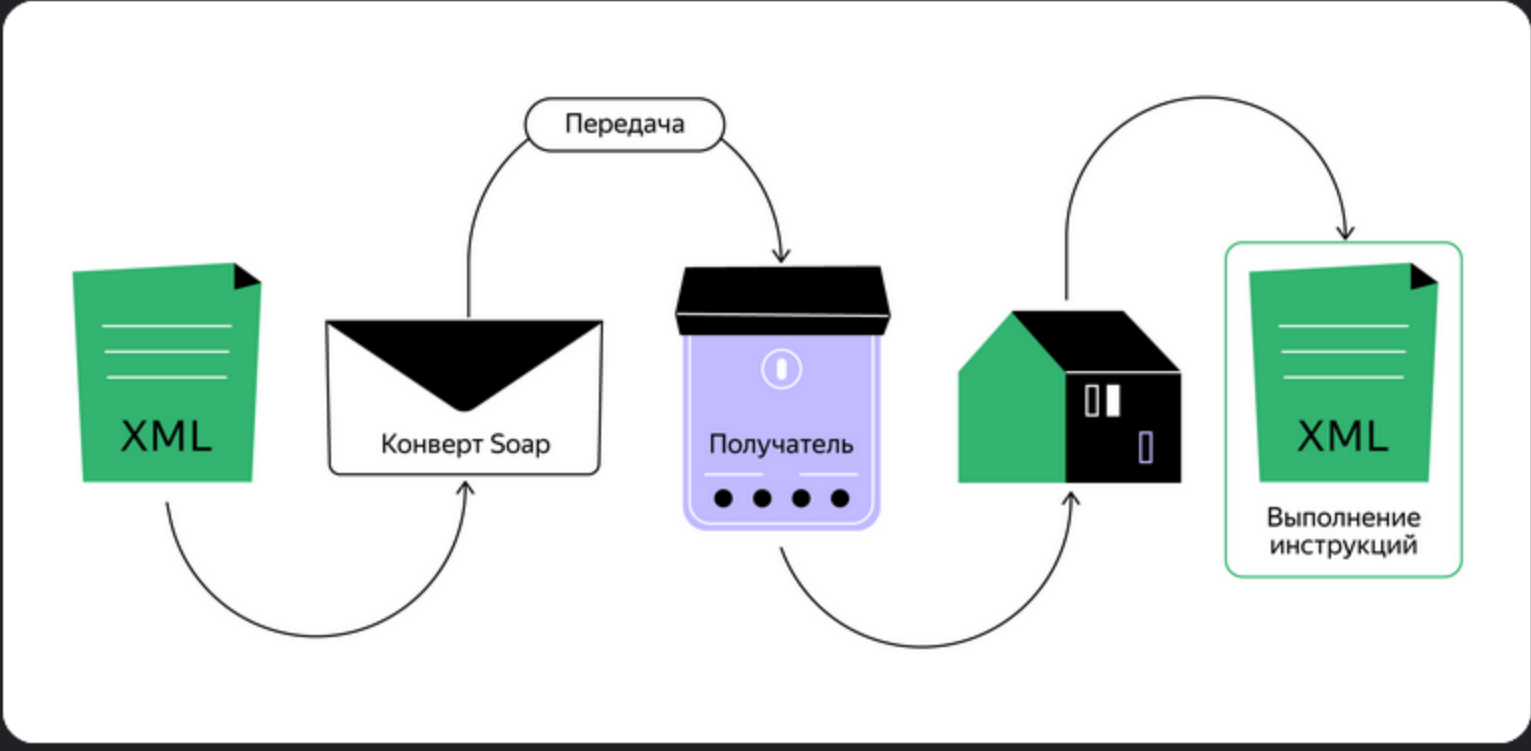
```
XML
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <!--Здесь служебная информация, например данные для авторизации-->
  </soap:Header>

  <soap:Body>
    <!--Здесь тело сообщения, пересылаемые данные-->
    <student>
      <username>Стас Басов</username>
      <faculty>Python</faculty>
    </student>

    <student>
      <username>Jacob Ludwig Karl Grimm</username>
      <faculty>Fable</faculty>
    </student>
  </soap:Body>

  <soap:Fault>
    <!--Здесь может быть список ошибок. Но это необязательно-->
  </soap:Fault>
</soap:Envelope>
```

Само сообщение в SOAP можно представить в виде обычного письма. Такая аналогия поможет лучше понять концепцию «конверта» SOAP.



### Язык WSDL: инструкция к веб-сервису

Практическая реализация API-сервиса на основе SOAP неразрывно связана с использованием языка WSDL.

Язык WSDL (англ. *Web Services Description Language*, «язык описания веб-сервисов») основан на XML и предназначен для создания машиночитаемых «инструкций», в которых описано, как программа-клиент должна взаимодействовать с сервером.

Для этого в файле с расширением *.wsdl* описываются детали взаимодействия клиента и сервера, в том числе адрес, куда нужно отправлять запросы, и структура конверта с запросом и ответом.

WSDL-файл — основа любого сервиса передачи данных, построенного на SOAP: программа-клиент должна знать адрес этого файла, и тогда она будет знать о веб-сервисе всё, что необходимо для работы с ним.

### Применимость SOAP

В настоящее время SOAP чаще всего применяется в сфере финансовых услуг, платёжных шлюзов, управления идентификацией — там, где требуется высокая надёжность передачи данных и хорошо документированный стандарт, на который можно было бы опираться в юридических документах.

### Архитектура REST

REST (англ. *REpresentational State Transfer*, «передача состояния представления») — это архитектурный стиль, набор принципов взаимодействия компьютерных систем, основанный на методах протокола HTTP. В отличие от SOAP, REST не подкреплён официальным стандартом.

Одно из основных понятий в REST — «ресурс». Это довольно абстрактное понятие: ресурсом в REST называют «всё, чему можно дать имя в структуре сервиса». Ресурсом может быть что угодно, к чему разработчик REST API считает важным предоставить доступ клиенту.

В качестве примера можно привести цитату из книги *RESTful Web Services* (Leonard Richardson, Sam Ruby, 2007):

«

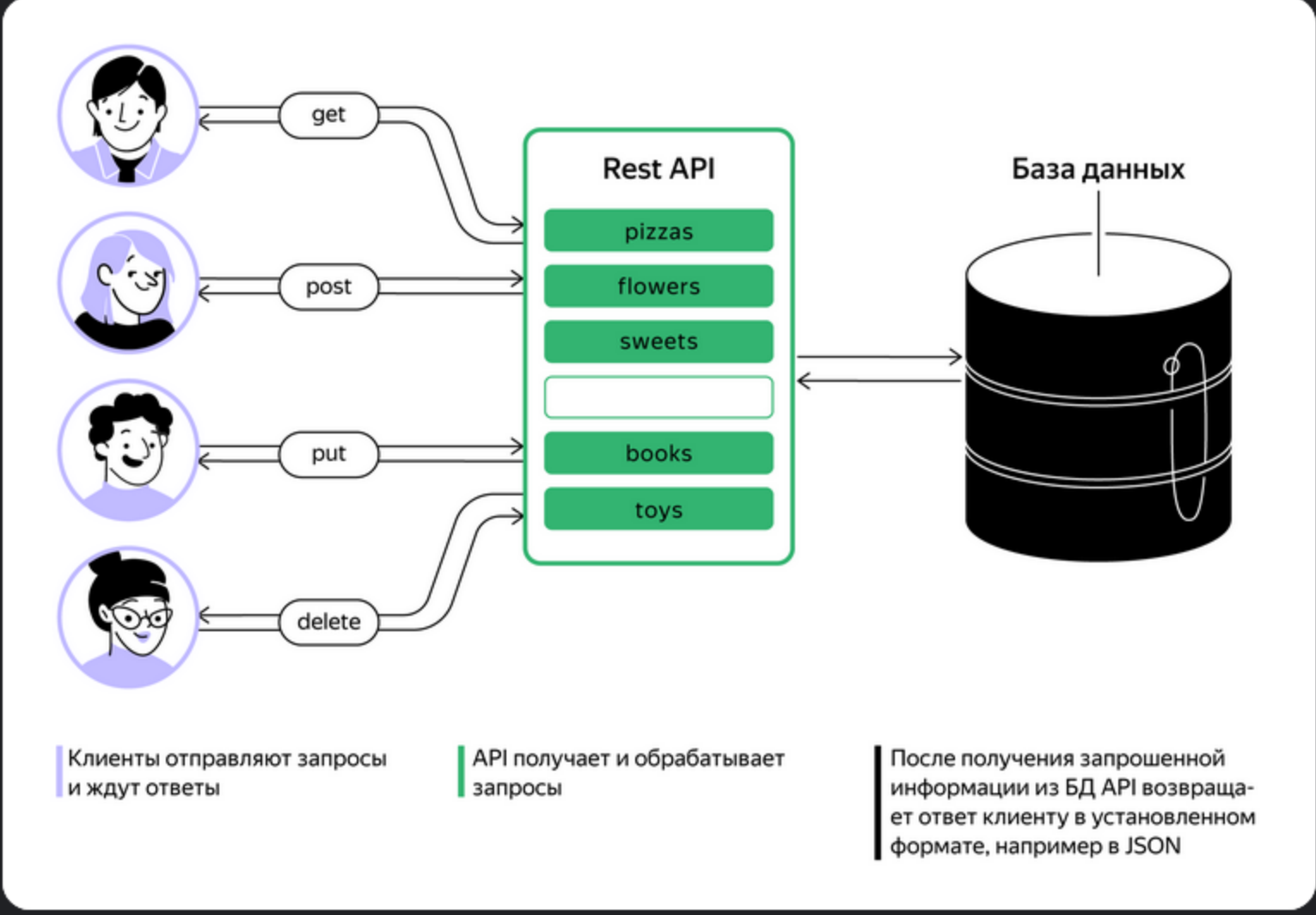
Вот несколько возможных ресурсов:

- Версия 1.0.3 компьютерной программы
- Последняя версия компьютерной программы
- Первая запись в блоге за 24 октября 2006 г.
- Дорожная карта Литл-Рока, Арканзас
- Некоторая информация о медузах
- Каталог ресурсов, относящихся к медузам
- Следующее простое число после 1024
- Следующие пять простых чисел после 1024
- Продажи номера для Q42004
- Отношения между двумя знакомыми — Алисой и Бобом
- Список открытых ошибок в базе данных ошибок

»

Вся логика работы REST API базируется именно на ресурсах: к API отправляются запросы, а из ответов клиент получает информацию.

На схеме — воображаемый сервис, который хранит информацию о пище, цветах, следстве, книгах и игрушках. Чтобы получить информацию о пище, нужно сделать запрос к ресурсу `/pizzas`, но конфет там не получишь: за конфетами — к ресурсу `/sweets`.



Архитектура REST допускает обмен данными в различных форматах, например в HTML, JSON, XML или даже в формате простого текста, в то время как строгий SOAP допускает только XML.

### Язык запросов GraphQL: служба одного окна

GraphQL — это не протокол и не архитектура, это язык запросов к API.

GraphQL был разработан в компании Facebook в 2012 году, а в 2015 году был выпущен в открытый доступ.

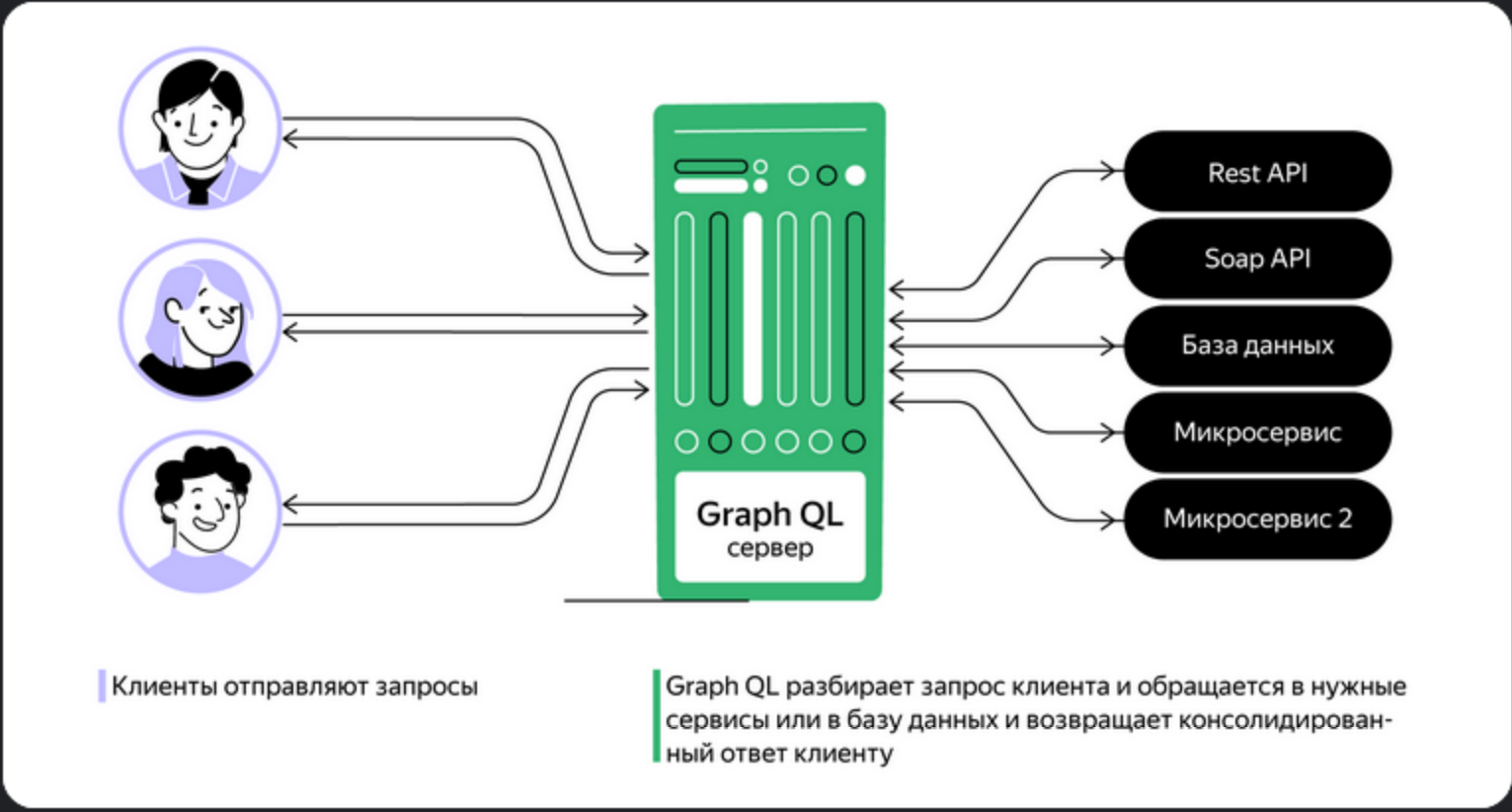
У GraphQL есть три основные особенности:

- он позволяет клиенту точно указать, какие данные ему нужны;
- облегчает агрегацию данных из нескольких источников (в отличие от REST API, который вернёт данные только одного ресурса);
- использует систему типов для описания данных.

Вернёмся к воображаемому API с пищей и другими полезными вещами.

Если API основан на архитектуре REST, голодный библиофил-сладкоежка для получения информации о пище, конфетах и книгах будет вынужден сделать три запроса к трём ресурсам.

В API, основанном на GraphQL, можно обойтись одним запросом: «Дайте мне информацию о большой „Маргарите“, об ирисках и ещё пришлите две главы из „Улисса“, пожалуйста!»



### Система удалённого вызова процедур gRPC: данные в бинарном виде

gRPC — это система удалённого вызова процедур, разработанная в Google в 2015 году.

В gRPC сообщения передаются в формате *protobuf* — это независимый от платформы и языка формат передачи данных, разработанный в 2008 году в Google. Основная специфика gRPC в том, что он оперирует данными в бинарном, а не в текстовом виде.

В gRPC клиентское приложение может напрямую вызывать метод серверного приложения на удалённом компьютере, как если бы это был локальный объект.

Идеальный вариант использования gRPC — это общение между микросервисами и построение коммуникации между мобильным приложением и сервером.

### Самый лучший подход для API

У каждого варианта реализации API-сервиса есть свои особенности. Выбор технологии зависит от требований к проекту. Может быть важен язык программирования или формат, в котором передаются данные; могут быть заявлены требования к быстрдействию или объёму передаваемых данных; возможно, есть необходимость опираться на строгий стандарт; даже квалификация разработчиков может быть аргументом в пользу выбора той или иной технологии.

Программисты и архитекторы API взвешивают все плюсы и минусы и выбирают вариант, наиболее подходящий конкретному проекту. Лучшая технология — та, которая лучше всего ляжет на требования к вашему проекту.

Далее в курсе вы будете работать именно с REST API — это популярная и востребованная на рынке архитектура; изучение REST даст навыки, которые впоследствии помогут разобраться и с другими технологиями.