

A close-up, high-contrast photograph of a gorilla's face, showing its eyes, nose, and mouth. The gorilla has a serious expression. The image is dark and textured, with the gorilla's fur appearing as a complex pattern of brown and black. The lighting is dramatic, highlighting the gorilla's features against a dark background.A solid orange horizontal bar.

Planet of the APIs

Ng Pan Yong
Chief Cloud Engineer
@pyx7b

title: Table of Contents

chapters:

- title: What are APIs

subChapters:

- title: Evolution of APIs

- title: OS API

- title: RPC

- title: Rise of the Web APIs

subChapters:

- title: SOAP

- title: REST

- title: The Others: WebSocket, GraphQL, gRPC

- title: Working with REST API

subChapters:

- title: OpenAPI with Swagger

- title: curl

- title: postman

- title: API Rules the World

subChapters:

- title: The War of Stacks

- title: NodeJS

- title: FAST API

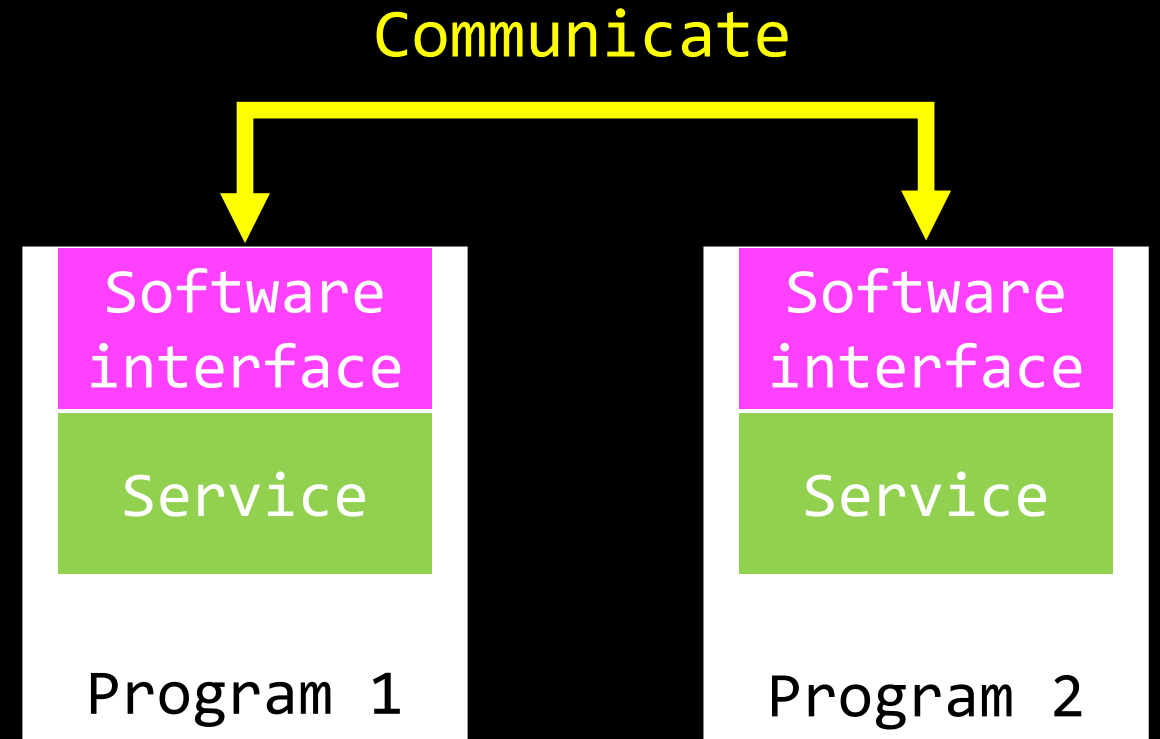
- title: In the End – United by API



What are APIs

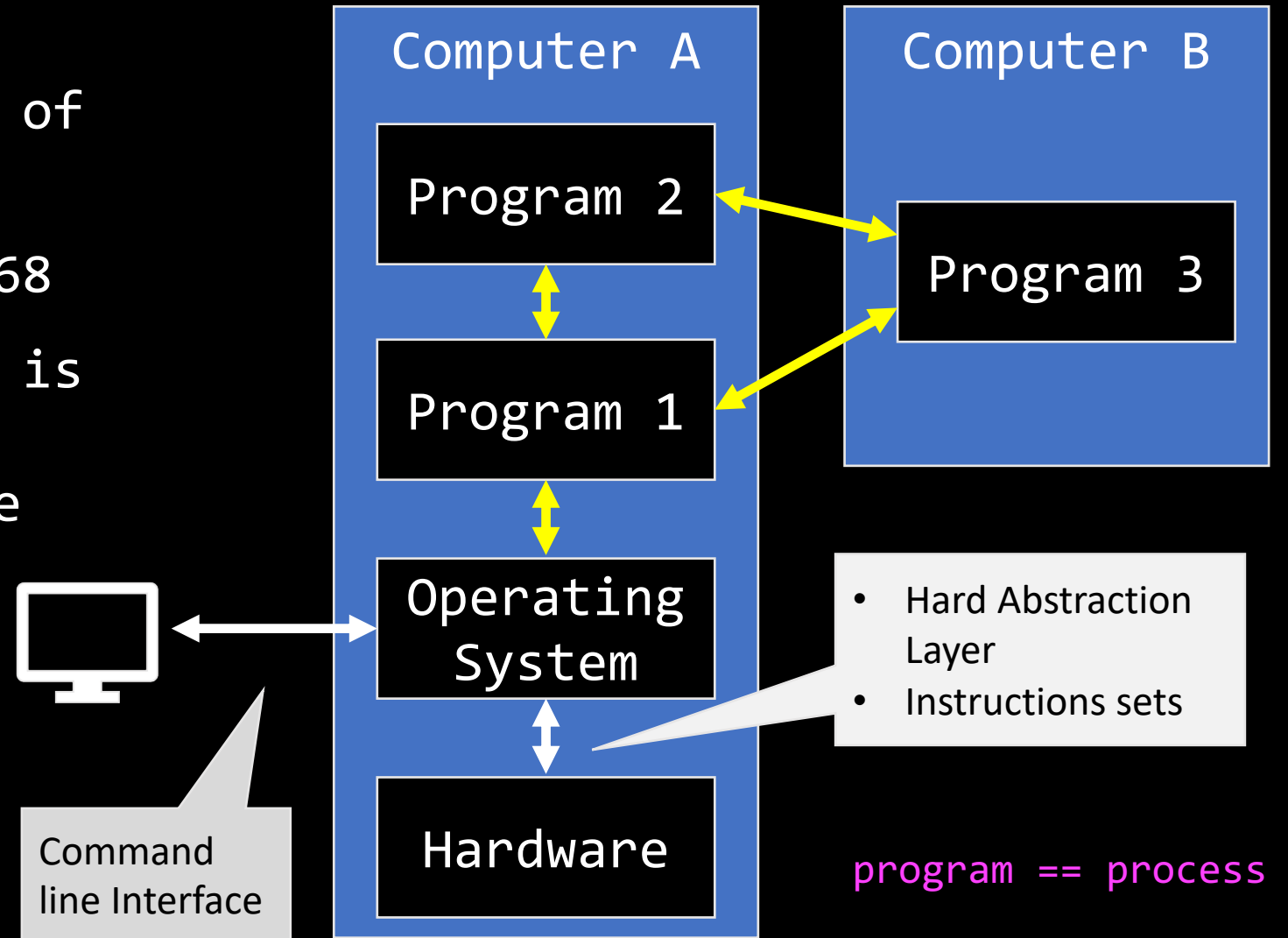
Wikipedia:

An application programming interface (API) is a way for two or more computer programs to **communicate** with each other. It is a type of **software interface**, offering a **service** to other pieces of software.



Evolution of APIs

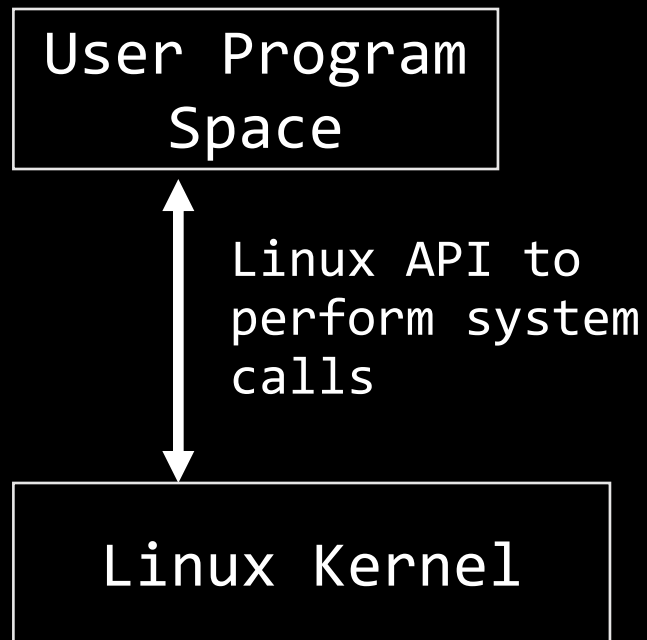
- The concept of API was conceived since the time of sub-routines
- Term was conceived in 1968
- At the lowest level APIs is the mechanism for interfacing with hardware
- Goals of API:
 - abstraction
 - standardization
 - interoperability



Operating Systems APIs

OS Level level APIs, such as Linux API and Win32 API, provide programs with interfaces to access system resources & core services such as the file system, device drivers, network interfaces, and other resources provided by the kernel.

program == process



```
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>

int main() {
    pid_t child_pid = fork();

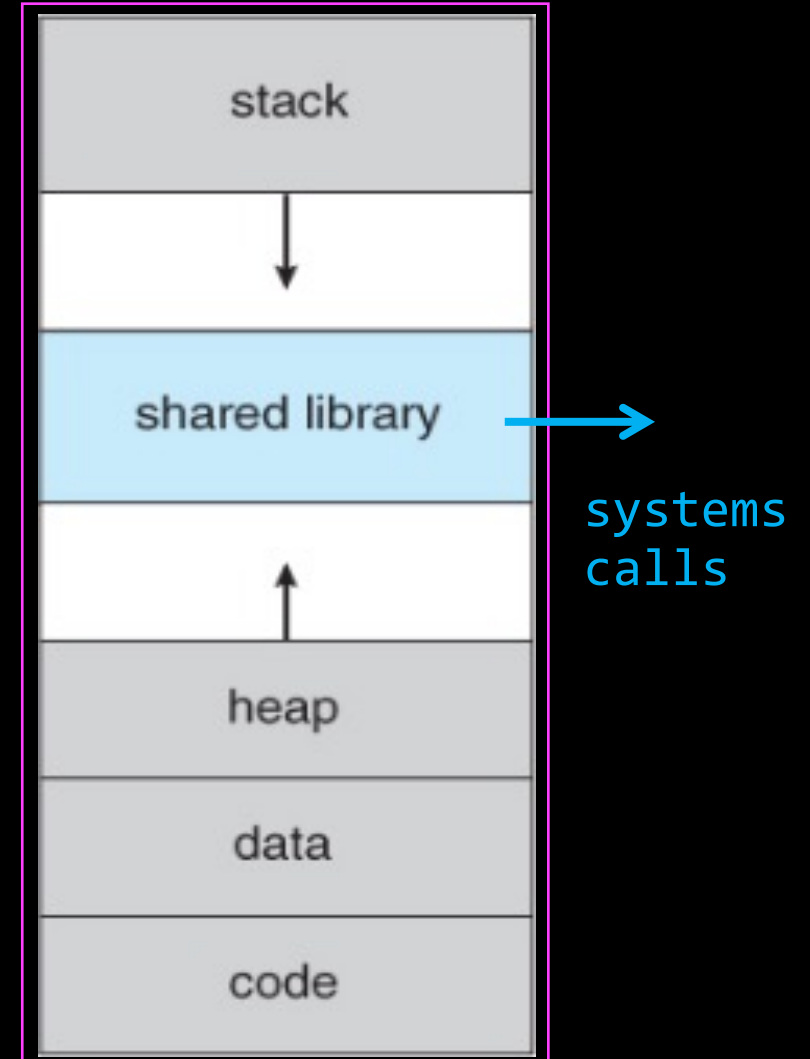
    if (child_pid > 0) {
        printf("Child process ID: %d\n", child_pid);
        wait(NULL);
        printf("Child process finished\n");
        exit(EXIT_SUCCESS);
    }
    else {
        exit(EXIT_FAILURE);
    }
}
```

Libraries are not APIs

- In **Windows** **DLL** (Dynamic Linked Libraries) are **compiled code** that can be loaded into the the user **program memory address space** to access OS API.
- The equivalent in **Linux** is **shared libraries** e.g. **libc**, **libssl**

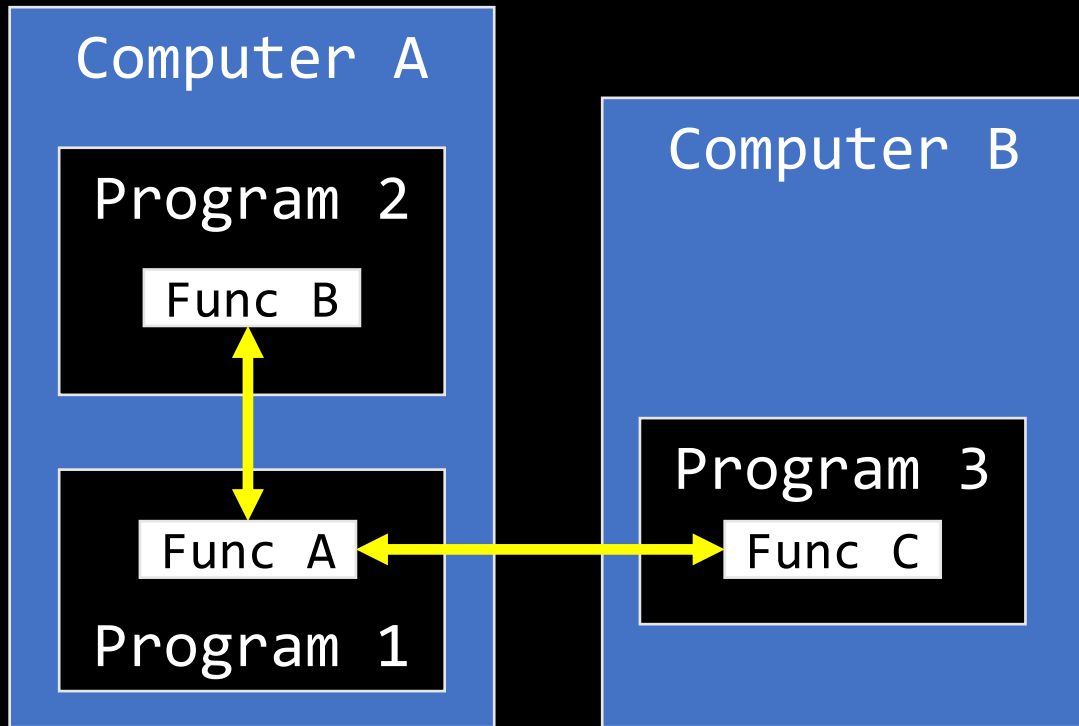
Link libraries are not considered APIs because they operate in the memory space of the program and it is the system calls that provides the **interface** to the underlying OS **services**

Program/Process
Memory Address Space



Remote Procedure Calls

Remote Procedure Calls (RPCs) are a mechanism that allows a computer program to **execute code** in a **different memory address space**, which may be on a remote machine. The basic idea behind RPCs is that a program can call a function or procedure in a different address space just as if it were a local function call.



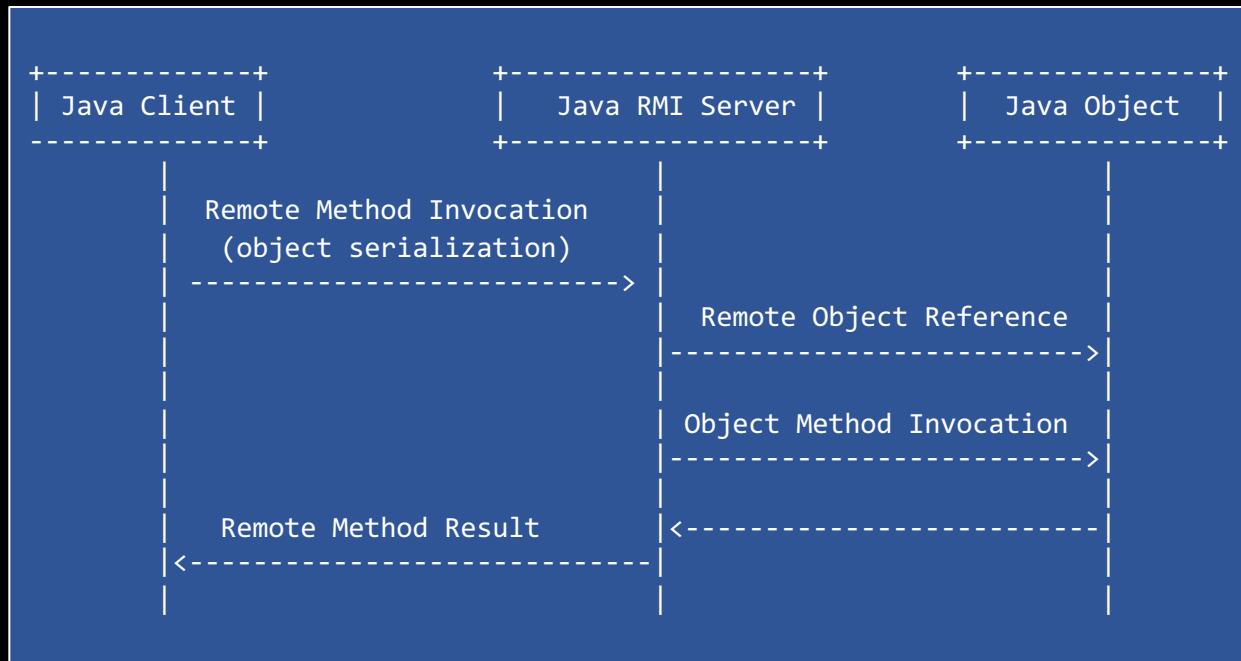
program == process

- Form of inter-process communication
- Can be used over different network protocols such as TCP, UDP, and HTTP
- Requires the data to be serialized and deserialized for transfer over the network
- Involves **IDL** (Interface Definition Language) for cross-platform integration. IDL define the methods, arguments, and return types of the remote procedure calls
- Examples for RPC: Java RMI, CORBA, Thrift, Avro, protobuf

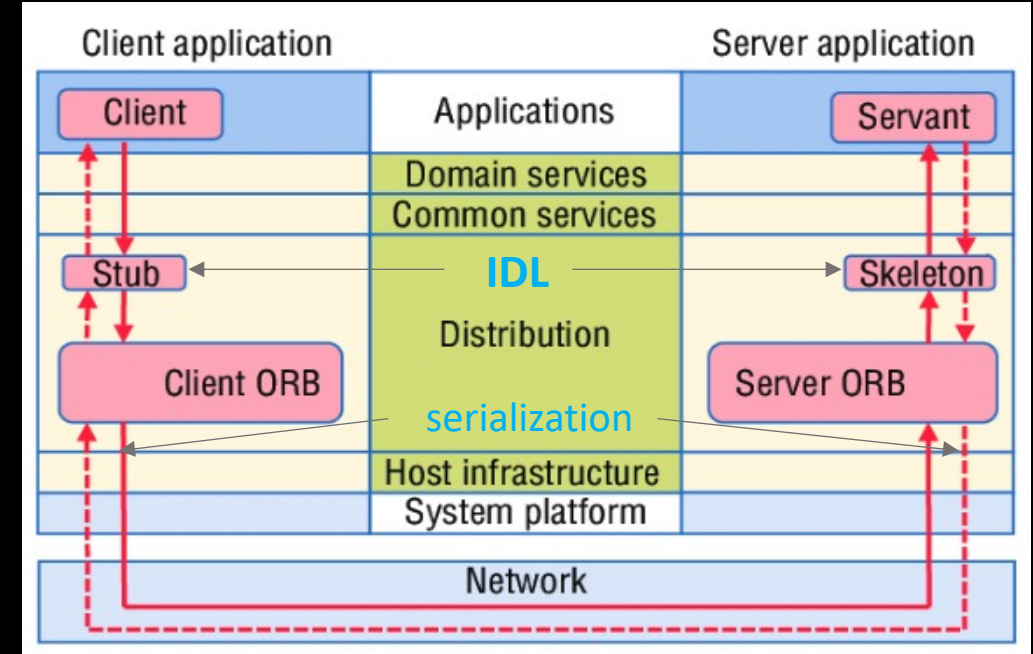
RPC Beware

- Requires client to implement complex error handling mechanisms
- Highly susceptible to the availability of a network and the remote service
- Performance is highly dependent on the network latency between client and server
- Compatibility issues with **programming language** and IDLs

JAVA RMI



CORBA



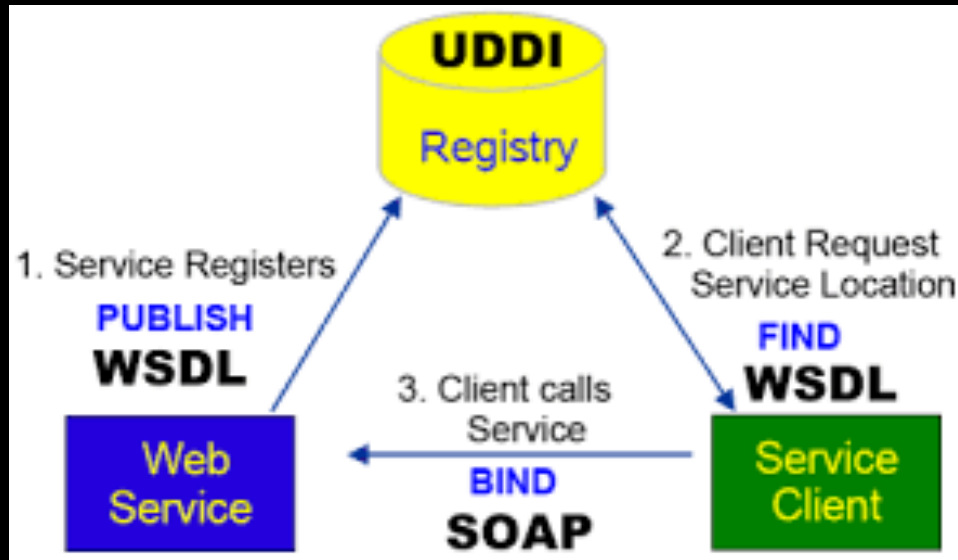
RISE

of the **Web APIs**



First Came SOAP Web Services

- **Good**: over standard HTTP protocol
- **Bad**: dependant on XML which adds overhead including transformation & parsing.
- **Bad**: WSDL like IDL, laborious to maintain
- **Ugly**: UDDI global registry of web service concept failed to take-up



```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="Bookstore"
  targetNamespace="http://example.com/bookstore"
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:tns="http://example.com/bookstore"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <types>
    <xsd:schema
      targetNamespace="http://example.com/bookstore">
      <xsd:element name="bookId" type="xsd:int"/>
      <xsd:element name="title" type="xsd:string"/>
      <xsd:element name="author" type="xsd:string"/>
      <xsd:element name="price" type="xsd:float"/>
    </xsd:schema>
  </types>

  <message name="getBookRequest">
    <part name="bookId" element="tns:bookId"/>
  </message>

  <message name="getBookResponse">
    <part name="title" element="tns:title"/>
    <part name="author" element="tns:author"/>
    <part name="price" element="tns:price"/>
  </message>

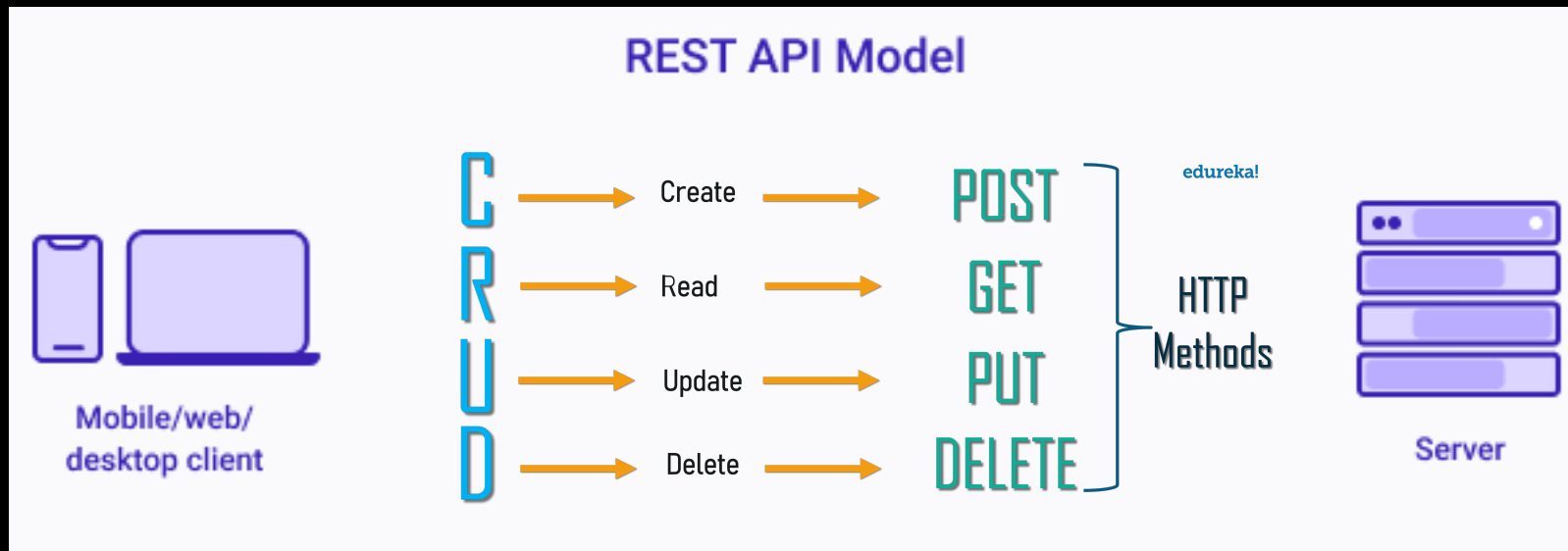
  <portType name="BookstorePortType">
    ...
```

Then Came RESTful API

- Uses a **simple** and **platform independent** set of HTTP methods interact with resources identified by **URIs**
- **Lightweight**: use of JSON and scaled to handle large amounts of traffic.
- **Stateless**: simplifies server-side development and allows for better **scalability** and **fault tolerance**

The term "REST" stands for "Representational State Transfer", which was coined by Roy Fielding in his doctoral dissertation in 2000

"Representational State Transfer" reflects the idea that a client can access and manipulate the representation of a resource (i.e., its state) through a standardized set of operations (i.e., transfer).



REST Example: Pet store

GET **/pet/findByStatus** Finds Pets by status

```
curl -X 'GET'
'https://petstore.swagger.io/v2/pet/findByStatus?status=
available' \
-H 'accept: application/json'
```

GET **/pet/{petId}** Find pet by ID

POST **/pet** Add a new pet to the store

```
curl -X 'POST' 'https://petstore.swagger.io/v2/pet' \
-H 'accept: application/json' \
-H 'Content-Type: application/json' \
-d '{
  "id": 0,
  "category": { "id": 0, "name": "string" },
  "name": "doggie",
  "status": "available"
}'
```

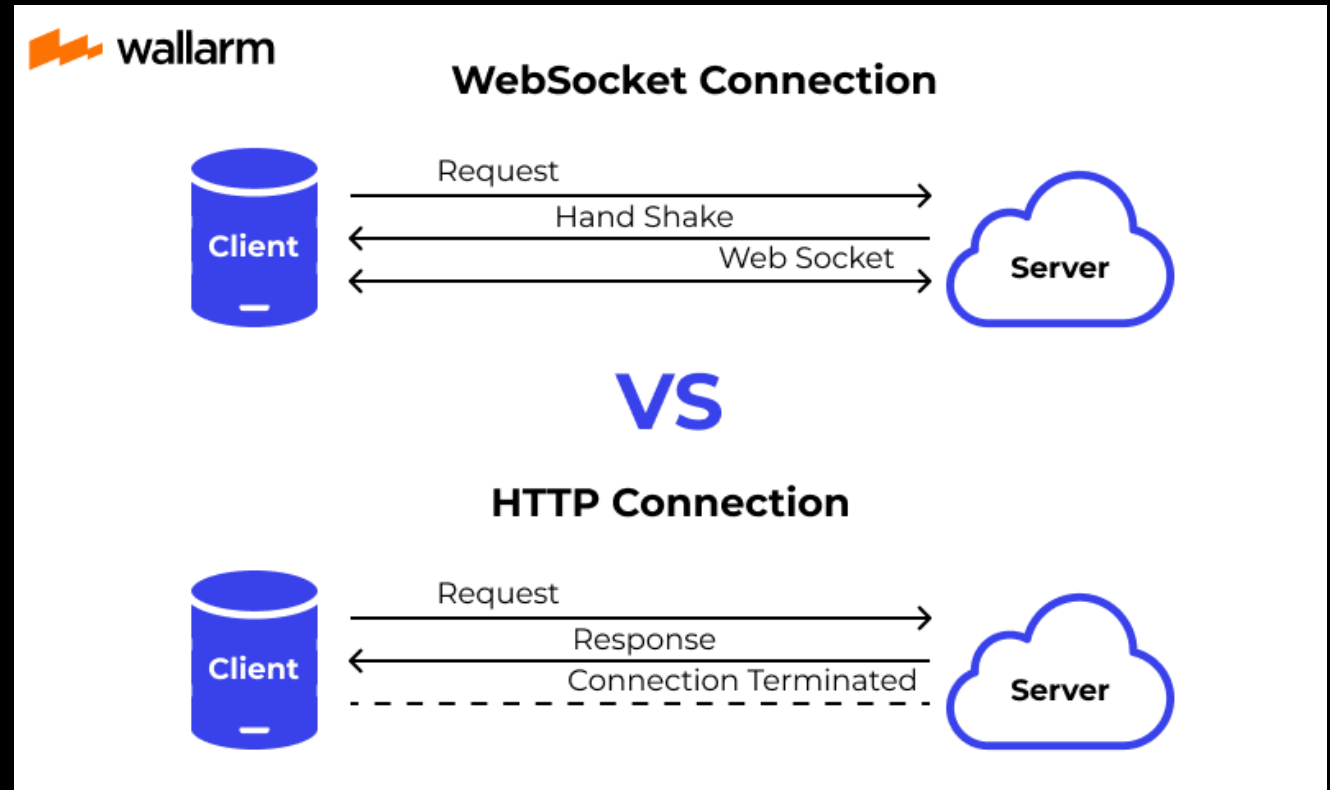
PUT **/pet** Update an existing pet

DELETE **/pet/{petId}** Deletes a pet

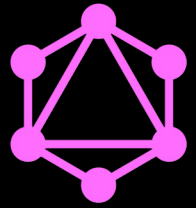
<https://petstore3.swagger.io>

Going 2-way: Web Socket

- Enable two-way communication between a client and a server over a single, long-lived connection.
- For real-time applications
- Allow binary data transfer
- Supported in most modern web browsers using the WebSocket API, which is part of the HTML5 specification.



>> Do not be confused with web hook which is a one-way server-to-client typically for notifications



GraphQL

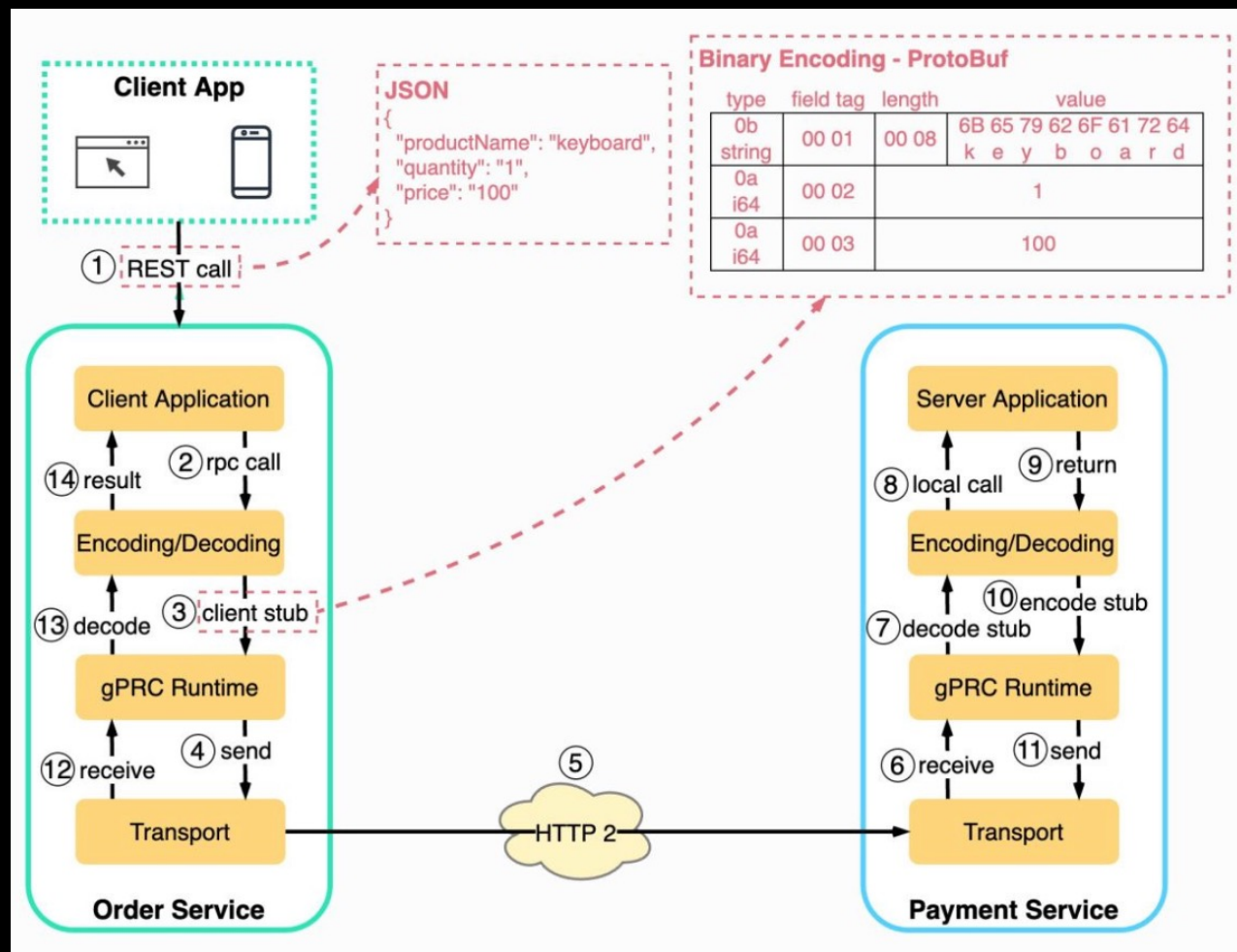
- GraphQL is an open-source query language and runtime for APIs developed by Facebook.
- Defines a **schema** that provides a strongly-typed interface for client
- Unless REST, allow clients to **specify** exactly what data they need
- **Aggregate** multiple resource requests in a single query
- **Subscriptions**: for client to receive notifications for data modifications
- **Mutations**: apply data modifications to resources

```
query {  
  pets(species: "Cat", age_lt: 3) {  
    id  
    name  
    age  
    breed  
  }  
}
```


gRPC

RPC is dead, Long Live gRPC

- gRPC is a high-performance, open-source framework originally developed by Google.
- Communicate over **HTTP/2** or TCP, using a binary serialization format called Protocol Buffers.
- More commonly used for communication between micro-services
- Some adoption on mobile client



Working with REST APIs



API Design Principles

R esilient	Stateless distributed computing and incorporate error handling, time-outs and monitoring.
A bstraction	Course grain API that provides functionality with abstraction away implementation details
M aintainable	Version control, flexible to change, keep things simple, intuitive and well-documented.
P erformant	Loose-coupling and asynchronous communication to improve the responsiveness and scalability of the API. Use caching to reduce network round-trips.
S ecured	Encrypted communication, authentication & authorisation, data validation, security best practices.

Use JSON

JSON stands for JavaScript Object Notation. It is a lightweight data interchange format that is easy for humans to read and write, and easy for machines to parse and generate. it is language-independent.

Useful Links:

<https://quickref.me/json.html>

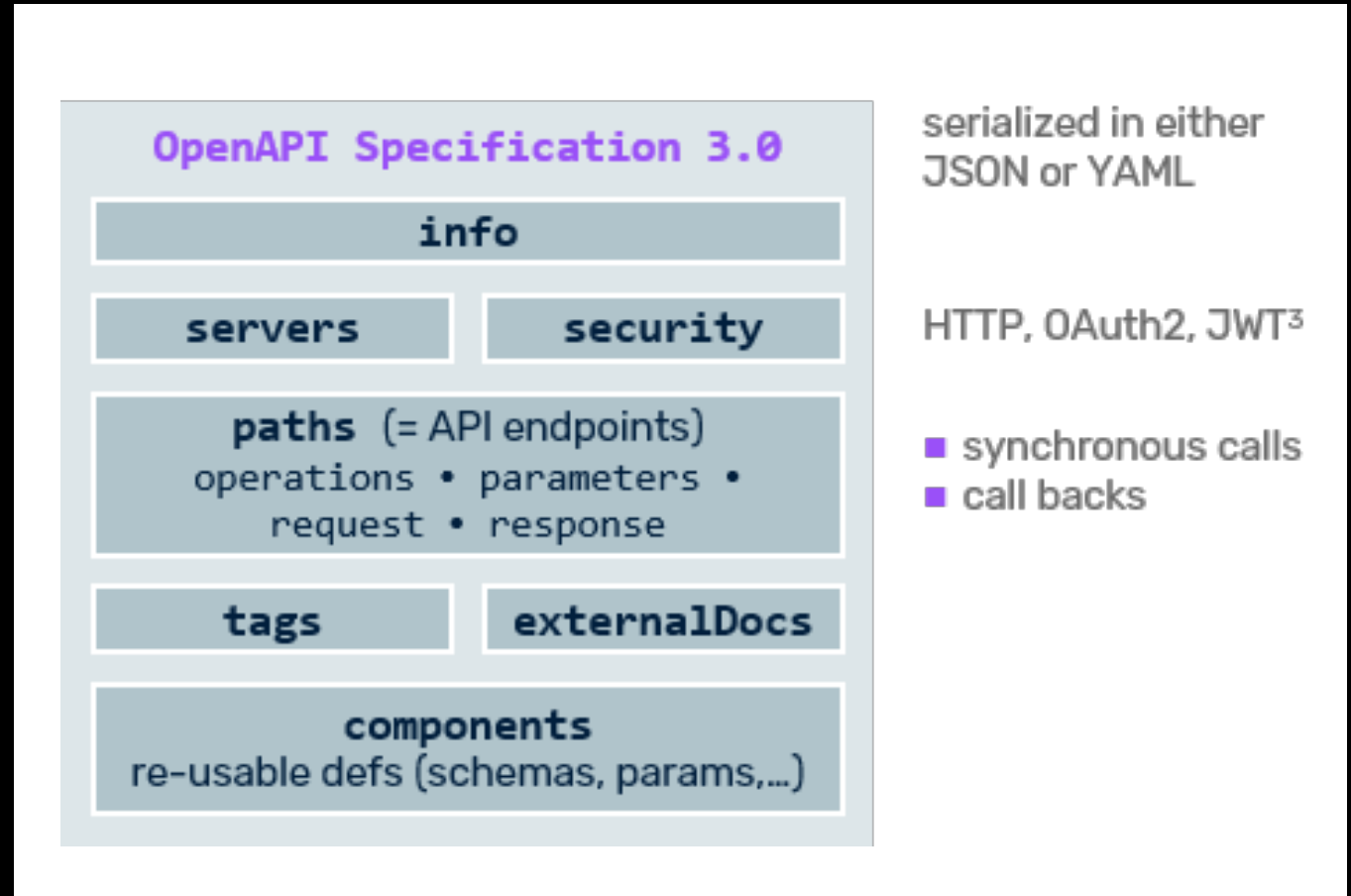
<https://jsoneditoronline.org>

<https://codebeautify.org/jsonviewer>

```
{
  "employees": [
    {
      "Name": "John Doe",
      "Active": true,
      "age": 30,
      "jobTitle": "Software Engineer"
    },
    {
      "Name": "Jane Smith",
      "Active": true,
      "age": 25,
      "jobTitle": "Web Developer"
    },
    {
      "Name": "Bob Johnson",
      "Active": true,
      "age": 40,
      "jobTitle": "Project Manager"
    }
  ]
}
```

Specify with OpenAPI/Swagger

- Open-source specification for building, documenting, and consuming RESTful APIs.
- Host API on SwaggerHub
- Online Editor
- Download APIs, Documentations and SDKs
- VSCode Extension from 42crunch for API Security



Know curl://

`curl` is a command-line tool used to transfer data to or from a server using various protocols, including HTTP, FTP, SMTP, POP3, and many others.

hide progress	verbose	extra info	output	timeout
-s	-v --trace-ascii <file>	-w "format"	-O -o <file>	-m <seconds>
POST	POST encoded	multipart formpost	PUT	HEAD (ers too)
-d "string" -d @file	--data-urlencode "[name]=val"	-F name=value -F name=@file	-T <file>	-I -i
custom method	read cookiejar	write cookiejar	send cookies	user-agent
-X "METHOD"	-b <file>	-c <file>	-b "c=1; d=2"	-A "string"
proxy	add/remove headers	custom address	smaller data	insecure HTTPS
-x <host:port>	-H "name: value" -H "name:"	--resolve <host:port:addr>	--compressed	-k
Basic auth	follow redirects	parallel	generate code	list options
-u user:passwd	-L	-Z	--libcurl <file>	--help

<https://quickref.me/curl>

Powershell (pwsh)

```
PS C:\> Invoke-WebRequest
```




Postman

- www.postman.com
- Great tool for testing APIs
- Sign-up for free
- Download Clients

What is Postman?

Postman is an API platform for building and using APIs. Postman simplifies each step of the API lifecycle and streamlines collaboration so you can create better APIs—faster.



API Tools

A comprehensive set of tools that help accelerate the API Lifecycle - from design, testing, documentation, and mocking to discovery.



API Repository

Easily store, iterate and collaborate around all your API artifacts on one central platform used across teams.



Workspaces

Organize your API work and collaborate with teammates across your organization or stakeholders across the world.



Governance

Improve the quality of APIs with governance rules that ensure APIs are designed, built, tested, and distributed meeting organizational standards.



API Rules the World

The War of Stacks

LAMP Stack



OSS, CMS,
Web2.0

MEAN Stack



Express



AJAX, Mobile responsive,
Single Page Applications

MERN Stack



Express



JAMstack



{ api }



Headless
CMS

NodeJS & Express

- **NodeJS** is an open-source, cross-platform server based on JavaScript runtime. It uses an event-driven, non-blocking I/O model that allows it to handle multiple requests simultaneously with minimal resources.
- **Express** is Web Framework that runs on Node.

Get Started

1. Download and install Node
2. Ask ChatGPT to create an express API server that has 1 get method for retrieving pet by id and 1 post method to add new pet
3. Create project
4. Install NPM packages
5. Cut and paste code
6. Start Server

FastAPI

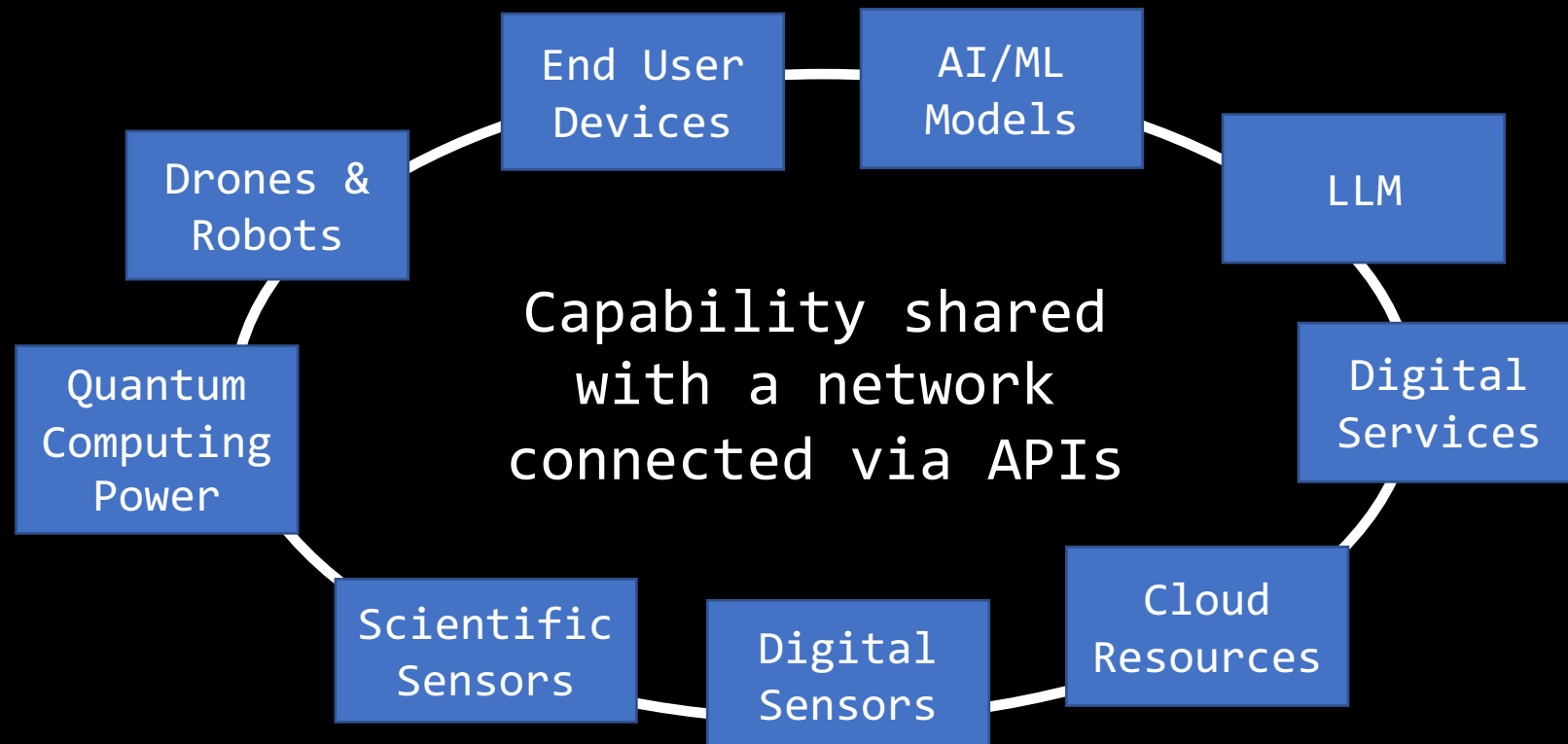
FastAPI is a modern, fast (hence the name), web framework for building APIs with Python. It is designed to be easy to use, high-performance, and to provide automatic validation of request data & automatically generate OpenAPI documentation

Get Started

1. Create a project folder with virtual environment
2. Ask ChatGPT to write an api server using fast API for a book store with book name, author and description. 5 books preloaded
3. Run the code on uvicorn
4. Check out the docs
5. Call OpenAI API

In the End: United by AI

- APIs provides abstractions to allow different technologies to be integrated, consuming services to create multiplying effect
- Shift thinking to – “How can others consume my service?”





Thank You

<https://github.com/htxsg/citizendeveloper/tree/main/planetOfAPIs>