Introduction to Web Services

Ahmad Hamo
2nd Semester 2024-2025

The Web as it was

- Designed for people to get information
  - Focuses on visual display (as in HTML)
  - Lacks support for meaning
  - Weak Interactivity and Reusability
- Supports low-level interactions
  - HTTP is stateless
  - Processing is client-server
  - Creates avoidable dependencies among what should be independent components

But, Not easy to program

The Web as it is becoming

- Enable interactions autonomous, heterogeneous parties (information providers and users)
  - Go beyond visual display to capture meaning → **Semantic Web**
  - Support standardized interfaces → **Web services**
  - Support complex activities → **processes**
  - Support rich interactions among autonomous parties → **agents**

3

What is an API

An API is where you tell a program you don't own to run.

- **A** stand for application, **P** for programming, and **I** for interface.
- You can think of an API conversely. ***The interface (I) is where you tell a program (P) to run in an application (A).***
- A simple example of this is Google.
  - The browser or webpage is the interface.
  - The program is the search function.
  - The application is Google.
- When referring to APIs, people usually mean ones that exist where the server is located.

4

What is an API?

Letter	Word	Meaning
A	Application	Software that does a task
P	Programming	Program (P) that does the task in the Application (A)
I	Interface	Place (I) to tell the program (P) to run

Level of Difficulty	Example	I (Interface)	P (Program)	A (Application)
Simple	Viber message using cell phone	Cell phone	Messaging	Viber
Simple	Google search using computer	Computer	Search	Google
Moderate/Complex	Create orders in eBay when you get them (no browser)	eBay	Create order	eBay

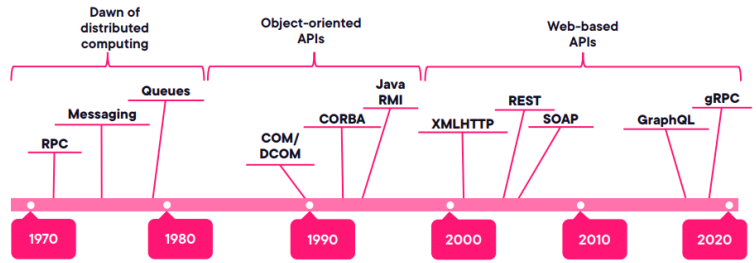
An API exists where you can tell (I) a computer program (P) to run in an application (A)

What makes APIs great?

- Just use the program don't write it!
- Platform independent
- Upgrade safe

5

History of (Web) APIs



6

Web Service - definition?

- A piece of business logic accessible via the Internet using open standards (Microsoft)
- Encapsulated, loosely coupled, contracted software functions, offered via standard protocols (DestiCorp)
- **A set of interfaces providing a standard means of interoperating between different software applications, running on a variety of platforms and frameworks (W3C)**

Our working definition: A service is functionality that can be *engaged*

7

What is a Web Service?

Web=internet
Service=api
Web Service=API that uses the internet

So, all web service is an API.
But not all APIs are web services. Not all APIs use the internet.

Web Services use:
- XML or JSON to format data over the internet
- REST, SOAP, or XML/RPC to transfer that data

8

Web Services Vs APIs

Aspect	Web Services	APIs (Application Programming Interfaces)
Purpose	Facilitate data exchange between software systems.	Provide access to the functionality or data of an application or service for developers.
Communication Protocol	Can use various protocols, including SOAP, REST, XML-RPC, JSON-RPC, and more.	Can use various protocols, often associated with RESTful APIs but not limited to them.
Data Format	Use multiple data formats, such as XML and JSON.	Can work with different data formats, including JSON, XML, or custom formats.
Interface	Do not have a user interface for direct human interaction.	Do not have a user interface but are used by developers to build applications with user interfaces.
Scope	A subset of APIs, specifically focused on web-based data exchange.	A broader concept encompassing various types of interfaces for software interaction.
Standards	WS-Security, WS-Policy for SOAP-based web services.	OpenAPI (formerly Swagger) for documenting RESTful APIs, GraphQL for querying APIs.

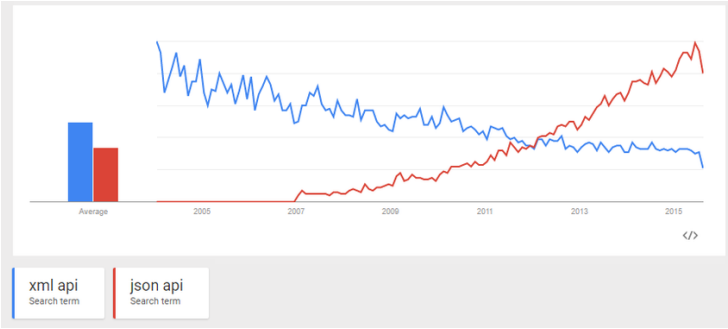
9

Web Services Vs Web Applications

Aspect	Web Services	Web Applications
Purpose	Facilitate data exchange between applications.	Provide services or perform tasks directly for end-users.
User Interaction	No user interface; meant for machine-to-machine communication.	Has a user-friendly interface for human interaction.
Data Exchange	Exchanges structured data between applications.	Involves user data input, processing, and result presentation.
Communication Protocol	Uses protocols like SOAP, REST, XML-RPC, or JSON-RPC.	Primarily uses HTTP/HTTPS for communication.
Security Focus	Focuses on securing data during transmission and access control.	Broader security aspects, including authentication, authorization, data validation, and protection against web vulnerabilities.
Examples	Payment gateways (e.g., PayPal API), weather data services.	Online banking, e-commerce (Amazon), email (Gmail), social networking (Facebook).

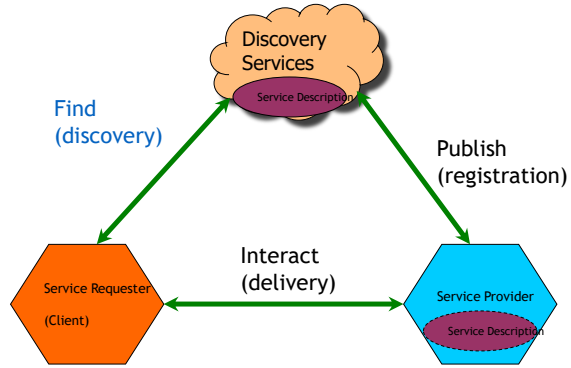
10

Using xml.api vs json.api – Google Trends



11

Web Services



12

Scope

Includes wherever Internet and Web technologies are employed

- **Internet**
- **Intranet**: network restricted within an enterprise
- **Extranet**: private network restricted to selected enterprises
- **Virtual Private Network (VPN):** a way to realize an intranet or extranet over the Internet

13

Service Composition (API Mashup)

- Is when you use several APIs to make your own API.
- Example : a travel website using the APIs of several airline companies to find the flight and price availability for the flight you search for. Using several APIs can be powerful.
- Obviously desirable and challenging
- But is this what we want?
  - Can or should implementations be hidden?
  - What about organizational visibility?
  - How to assess risk? How to handle exceptions?

14

Going Beyond the Idea of a Passive Object

- Passive Objects vs. Active Services : traditional object-oriented programming often treats objects as passive entities that respond to method calls.
- In contrast, RESTful services are active participants in a distributed system, capable of initiating actions, communicating with other services, and maintaining state (if needed).
- Proactive Behaviour : Services can act autonomously, such as triggering events, invoking call-backs, or pushing notifications to clients.

15

Applications of Composable Services

- **Portals**: integrates multiple services to provide a unified user experience.
- **Legacy system interoperation**: Modernizing an old banking system by wrapping its functionality in RESTful services.
- **E-commerce**: An online shopping platform that composes services for different functionalities.
- **Virtual enterprises**: A network of small businesses collaborating to form a virtual enterprise.
- **Grid computing**: A scientific research project that uses grid computing to process large datasets.

16

Autonomy

Independence of business partners (users and organizations):

Web services allow business partners to operate independently while still collaborating effectively

- **Political reasons:**
  - **Ownership of resources**: who controls the resources (data, infrastructure, etc.) that are exposed or consumed through the service.
  - **Control, especially of access privileges**: to ensure that only authorized users or systems can interact with the service.
  - **Payments**: Web services often involve financial transactions

- **Technical reasons:**
  - **Opacity of systems** with respect to key features, e.g., **precommit** in distributed databases:
    - Opacity refers to hiding the internal workings of a system from its users or consumers, ensuring they only see the desired functionality.
    - The **precommit phase** is a critical step in the **two-phase commit (2PC)** protocol, which ensures **atomicity** and **consistency** across multiple nodes.

Heterogeneity

- Heterogeneity in web services refers to the diversity of systems, technologies, platforms, and protocols

- Independence of component designers and system architects

- **Political reasons**
  - **Ownership of resources**: In heterogeneous environments, different entities may own and control the resources (data, infrastructure, etc.) exposed through web services.
- **Technical reasons**
  - **Conceptual problems in integration**: differences in how systems represent data, processes, and business logic.
  - **Fragility of integration**: they rely on multiple components working together seamlessly.
  - Difficult to guarantee behavior of integrated systems

Dynamism

- refers to the ability of systems to adapt, evolve, and respond to changes in real-time without requiring complete reconfiguration or downtime.

- Independence of system administrators
- Needed because the parties change
  - Architecture and implementation
  - Behavior
  - Interactions
- Make configurations dynamic to improve service quality and maintain flexibility

19

Locality: How to Handle the Above

- Reduce sharing of data and metadata to reduce inconsistencies and anomalies
- Reduce hard-coding, which reflects out-of-band agreements among programmers
  - Bind dynamically to components
  - Use standardized formats to express data
  - Express important knowledge as metadata
  - Use standardized languages to express metadata
- Relax consistency constraints
  - Obtain remote knowledge only when needed
  - Correct rather than prevent violations of constraints: often feasible

20