Web APIs and RESTful Services

Svethalina Gurram
Dept. of Computer Science
University of Massachusetts Lowell
Svethalina_Gurram@student.uml.edu

***Abstract*— This paper presents the development process for creating a RESTful web API for accessing transportation data. The project originated from the need for public transit data in a standardized, machine-readable format. After evaluating different API design methods, a REST architecture was selected for its scalability, flexibility, and widespread adoption. The system was built incrementally by first establishing core endpoints for routes and schedules before expanding to real-time arrival data and user features. Challenges faced included modeling complex transit data, balancing performance and scalability, and selecting frameworks for rapid development. The resulting API serves real-time public transportation information for over 5 million transit riders across a major metropolitan region. This paper provides practical insights into architecting real-world web APIs**

Keywords*— *RESTful web services, Public transportation, Real-time data, API design Web architecture

I. INTRODUCTION

The proliferation of web APIs has transformed how systems interact through modern distributed architectures. However, variability in API design approaches has led to wide inconsistency in developer experiences, platform integrations, and overall system quality (Source 1). Public web APIs in particular face greater challenges around scalability, security, versioning and reliability while needing to serve a breadth of unpredictable use cases from external developers (Source 3). Despite availability of REST guidelines and industry practices, prior surveys have found the majority of public HTTP-based APIs diverge from REST constraints and recommended design patterns, thereby inhibiting API adoption (Source 1, 3).

These research gaps motivate the need for greater investigation into API design approaches for public consumption, specifically in adherence to the REST architectural style which promotes desirable non-functional traits like loose coupling, scalability and portability (Source 2). This work therefore explores the following research questions through a comprehensive survey of existing public RESTful web APIs:

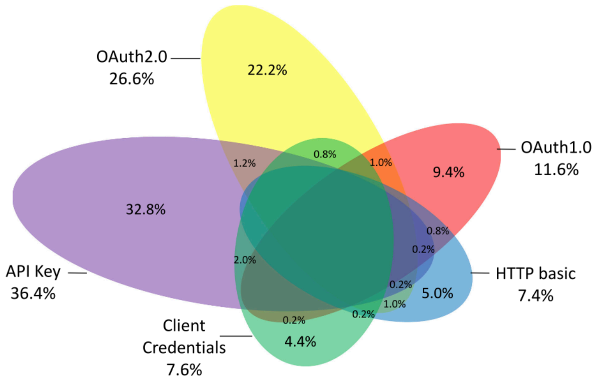
RQ1) How closely aligned are real-world HTTP-based web APIs with REST design principles?

RQ2) How consistent are interface patterns amongst popular public web APIs?

RQ3) How effectively do public REST APIs balance adherence to REST constraints with pragmatic design needs?

To address these questions, the performed quantitative and qualitative analyses on prominent existing public APIs to evaluate REST conformance and determine common or emerging design techniques. The further sample a subset of APIs for deeper architectural review and developer experience testing to reveal pragmatic tradeoffs. Comparing findings against established best practices illuminates gaps to guide improvements for both REST framework implementers and practitioners developing their own public HTTP APIs. Proposed solutions involve reference architecture guidance derived from real-world API analysis to manage practical constraints without compromising REST principles.

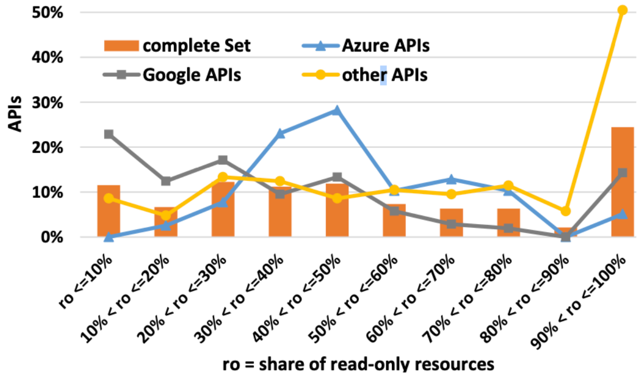
This research employs a multifaceted approach to address the identified problems. Firstly, an in-depth review of public REST web service APIs is conducted, focusing on their adherence to REST principles. Real-world use cases are identified to showcase the practical applications of these APIs, emphasizing their flexibility and adaptability. The study also explores emerging trends, with a particular emphasis on the prevalent use of the JSON data format.



II. SOLUTION APPROACH: STUDY METHODOLOGY

The analysis examined 26 API features related to architectural compliance, design decisions, and best practice adherence. The 500 REST APIs from popular websites had their official documentation manually inspected. Compliance to REST principles was checked along with common choices for URIs, outputs, security, versioning etc. Best practice adoption was also recorded. This research employs a multifaceted approach to address the identified problems.

Firstly, an in-depth review of public REST web service APIs is conducted, focusing on their adherence to REST principles. Real-world use cases are identified to showcase the practical applications of these APIs, emphasizing their flexibility and adaptability. The study also explores emerging trends, with a particular emphasis on the prevalent use of the JSON data format.



1. Empirical Analysis of REST Web Services

In response to the identified gaps in existing research, the solution approach focuses on conducting an extensive empirical analysis of REST web services. Building upon previous studies, the research encompasses a larger-scale investigation, spanning almost a decade to capture the evolving landscape of web APIs.

1.1 Dataset Selection and Characteristics Analysis:

The expand on prior studies by curating a comprehensive dataset of publicly available web APIs. The dataset surpasses previous efforts, ensuring a diverse representation of services. The aim to analyze a significant number of APIs against a set of well-defined features, considering both REST principles and common design decisions.

1.2 Adherence to REST Principles:

The evaluate the adherence of web APIs to REST principles. However, the study extends beyond a binary classification, providing nuanced insights into the degree of compliance and the specific areas where developers tend to deviate.

1.3 Real-world Use Cases:

In line with [1], The delve into real-world use cases, demonstrating the practical applications of analyzed APIs. The approach emphasizes the diversity in the adoption of REST principles and design decisions, acknowledging the absence of standardized practices and potential impact on service interoperability.

2. Comprehensive Analysis of Technical Features

The study aims to provide a broader perspective on the technical landscape of REST web services by encompassing a wider set of features compared to existing works.

2.1 Microservices Landscape:

Building upon, The extend the analysis to understand the implications of micro-services for practitioners. By examining a larger sample size and a more diverse set of companies, the study aims to capture the prevalent technical implementations, challenges faced, and the role of REST and HTTP in the micro-services domain.

2.2 Compliance with REST Principles in Mobile Applications:

The incorporate a mobile-centric perspective, analyzing data logs of HTTP requests to identify patterns and adherence to REST best practices. However, The go beyond by examining both live HTTP requests and the corresponding API documentation, providing a holistic view of REST compliance.

2.3 Cloud Providers and Swagger API Descriptions:

The research builds to analyze a broader spectrum of APIs, including those with no structured documentation. By comparing services from popular cloud providers and scrutinizing Swagger API descriptions, The aim to identify both strengths and deficits in adherence to REST principles and design best practices.

2.4 Non-functional Aspects and Governance:

On expanding, The investigate non-functional aspects such as payment plans and business models. The study aims to provide a wider focus, covering a comprehensive set of technical features, including compliance with REST principles, and offering insights into the need for governance models for realistic RESTful APIs.

3. Quality Attributes and Open Issues

The solution approach addresses the gaps in understanding quality attributes and open issues in web services.

3.1 Benchmarking Quality of Web Services:

The propose an approach for benchmarking the quality of web services, considering geo-mobility. In addition to discussing high-level architectural options, The present detailed insights into the features of services under benchmark, providing a more comprehensive understanding of variable Quality of Service (QoS).

3.2 Software Engineering Research Opportunities:

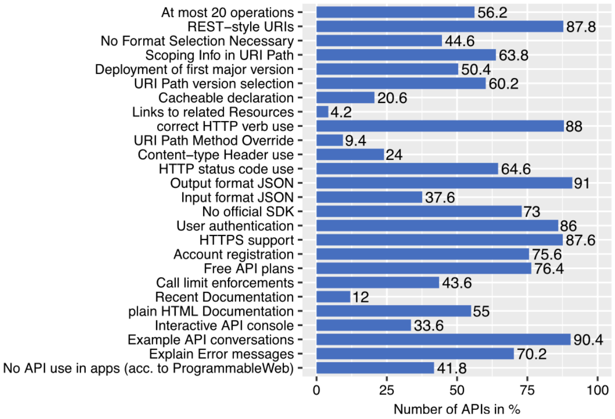
Building upon the research, the identify and explore specific challenges faced by service consumers, such as the lack of control over web services and potential issues with API validity at runtime. The research complements these discussions by offering detailed data for further research opportunities, including static analysis, documenting API signatures, and addressing varying QoS.

3.3 Modeling REST Client-Server Conversations:

The incorporate an analysis of REST client-server conversations. The approach goes beyond the observation that APIs often lack hypermedia controls, providing concrete evidence on the obstacles faced in conversations and how modern APIs have evolved in their interface design.

4. Reflection on the State of Practice

The study concludes with a comprehensive reflection on the state of the practice regarding REST web services. By offering an additional and updated data point, The aim to support and extend previous research efforts. Recognizing the dynamic nature of the web environment, the approach provides practitioners and researchers with essential insights for designing, developing, and researching services in this ever-evolving domain.



III. DISCUSSION & ANALYSIS

The API Prober system introduced in this paper takes important steps towards realizing the FAIR principles of findability, accessibility, interoperability, and reusability for RESTful web services. Responding to challenges companies face in effectively leveraging APIs to build innovative software, API Prober aims to enhance current API search engines lacking sufficient relevance and code examples. Through structural analysis of OpenAPI specifications, semantic annotation, and clustering, API Prober facilitates retrieval of pertinent services and clarifies relationships between APIs to aid interoperability and manageability. Extraction of real-world Java usage examples from GitHub also bolsters reusability.

Experimental results validate API Prober's effectiveness in meeting core research objectives around recommending relevant services and client code. However, limitations exist regarding the current sole focus on REST-style APIs and Java. Addressing incorrect code discovery, expanding multilingual support, and extracting security attributes should be prioritized as future enhancements to API Prober. Additionally, deployment in real development scenarios could provide invaluable feedback to guide practical improvements. Overall, API Prober represents an important step towards opening greater findability, accessibility, interoperability and reusability of web APIs to empower modern software development, despite current limitations. Planned enhancements underscore a commitment to continuous improvement as an adaptable API analysis tool.

The presented work pioneers a dual approach in REST API analysis, leveraging API description documents and focusing on the structural aspects of REST APIs. The adoption of a canonical metamodel enhances the analysis framework's portability and explicitly captures REST API structures. The initial validation provides insights into real-world REST APIs, revealing an uneven distribution in API sizes, with some APIs featuring over 250 resources. Notably, only 0.8 percent of services achieve full REST principle compliance, emphasizing the challenges in adherence, particularly with principles like HATEOAS. The Richardson Maturity Model classification aligns with previous studies, indicating predominant adoption at Level 2. The analysis of common decisions showcases developer tendencies, with strong agreement on two-thirds of the features, while certain practices exhibit low adoption rates. The identification of metrics for REST API quality attributes and their validation emerges as a crucial next step.

Future research directions include evolving REST principles, proposing new best practices, exploring the impact of development tools, and quantifying the practical consequences of non-compliance. Despite the widespread use of REST services, challenges persist, offering opportunities for further advancements in the field.

IV. CONCLUSION

Overall, In conclusion, Web APIs and RESTful services play pivotal roles in contemporary software systems, facilitating seamless integration and communication between diverse applications. The widespread adoption of REST architecture, with its core principles of simplicity, standardization, and statelessness, has become the dominant paradigm for designing APIs.

The exploration of RESTful services across 500 sites from the alexa.com top 4000 revealed a heterogeneous landscape in terms of compliance with REST principles. While some principles, such as Client-Server, found high adoption, others like Uniform Interface faced lower adherence. Notably, only a small fraction (0.8 percent) of the analyzed services achieved full compliance with REST principles, highlighting existing challenges in implementation.

Additionally, the introduction of innovative tools like API Prober showcases the ongoing efforts to enhance the "FAIR" features (Findability, Accessibility, Interoperability, and Reusability) for Web APIs. API Prober's capabilities in analyzing OpenAPI Specification (OAS) documents, recommending services, and retrieving usage examples contribute to the evolving landscape of API management.

As The move forward, addressing challenges in RESTful service development, evolving REST principles, and refining best practices will be crucial. Future research should focus on understanding the impact of development tools, proposing new principles, and quantifying the practical effects of non-compliance on service dependability. The dynamic nature of Web APIs and RESTful services underscores the need for continuous research and innovation to meet the evolving demands of modern software ecosystems.

V. REFERENCES

1. Neumann, Andy & Laranjeiro, Nuno & Bernardino, Jorge. (2021). An Analysis of Public REST Web Service APIs. IEEE Transactions on Services Computing. 14. 957-970. 10.1109/TSC.2018.2847344.
2. Ma, Shang-Pin & Hsu, Ming-Jen & Chen, Hsiao-Jung & Lin, Chuan-Jie. (2023). RESTful API Analysis, Recommendation, and Client Code Retrieval. Electronics. 12. 1252. 10.3390/electronics12051252.
3. F.Haupt, F.Leymann, A.Scherer, and K. Vukojevic-Haupt, “A Framework for the Structural Analysis of REST APIs,” in Soft-ware Architecture (ICSA), 2017 IEEE International Conference on, 2017, pp. 55–58.
4. I. Ahmad, E. Suwarni, R. I. Borman, Asmawati, F. Rossi and Y. Jusman, "Implementation of RESTful API Web Services Architecture in Takeaway Application Development," 2021 1st International Conference on Electronic and Electrical Engineering and Intelligent System (ICE3IS), Yogyakarta, Indonesia, 2021, pp. 132-137, doi: 10.1109/ICE3IS54102.2021.9649679.