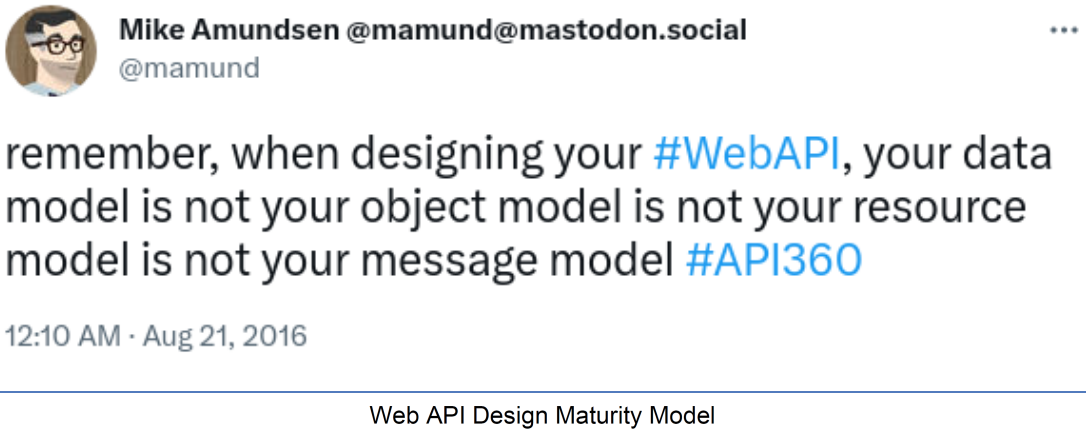
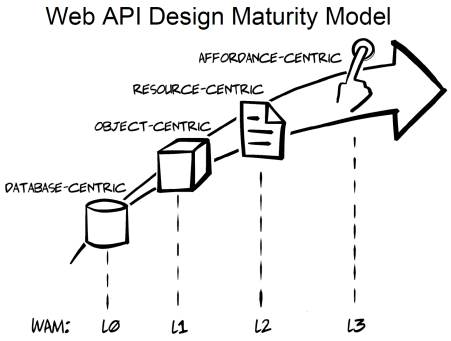
Amundsen’s Maxim

An open admonishment to all API architects, designers, & implementors

"Remember, when designing your Web API, your data model is not your object model is not your resource model is not your message model." — **Mike Amundsen**



The first public tweet of the maxim.



Web API Design Maturity Model — This talk (from 2016) contains one of the earliest public versions of the maxim.

What is Amundsen's Maxim?

Amundsen's Maxim is a principle for designing RESTful APIs that was first proposed by Mike Amundsen in 2012. The maxim states that:

> "Remember, when designing your Web API, your data model is not your object model is not your resource model is not your message model."

The maxim emphasizes that different layers of an API should be designed independently rather than tightly coupled.

The **data model** represents how information is stored (e.g., in a database), the **object model** defines how that data is structured in application code, the **resource model** determines how clients interact with the API (e.g., RESTful resources or hypermedia affordances), and the **message model** dictates how data is exchanged over the network (e.g., JSON responses, events).

Treating these as separate concerns enables better flexibility, maintainability, and evolvability, preventing rigid APIs that break when underlying implementations change. This principle aligns with REST, hypermedia, and API affordance-driven design, ensuring APIs serve client needs rather than mirroring backend structures.

What It Means

This maxim reminds API designers that different models serve different purposes and should not be conflated. Breaking it down:

Data Model ≠ Object Model

- The **data model** represents how data is structured in a database (e.g., tables, relations, normalization).
- The **object model** is how that data is represented in code (e.g., classes, objects, inheritance).
- A direct 1:1 mapping between these can lead to tight coupling and inflexibility.

Object Model ≠ Resource Model

- The **resource model** is how entities are exposed via the API (e.g., REST resources, hypermedia affordances).
- Mapping objects directly to resources often leads to rigid, CRUD-based APIs that don't evolve well.
- Resources should be designed around use cases, not internal object structures.

Resource Model ≠ Message Model

- The **message model** defines how data is transmitted between client and server (e.g., JSON payloads, event messages, HATEOAS links).
- APIs should be designed for network efficiency and interoperability rather than exposing raw resource representations.

Where It Comes From

Mike Amundsen, a REST advocate and hypermedia proponent, formulated this as part of his guidance on resource-oriented API design. It aligns with principles from REST, Domain-Driven Design (DDD), and API Evolution.

The idea echoes Eric Evans' DDD principle that different contexts (Bounded Contexts) demand different models, and also Fielding's REST constraints, which emphasize abstraction between client and server.

Why It's Important

Prevents Rigid, Overly Coupled APIs

- If APIs directly reflect the database schema, any database change breaks the API.
- If APIs expose objects as-is, it leads to tight coupling between frontend and backend.

Encourages API Design Based on Affordances

- A resource model should focus on actions and workflows (what clients need), not on mirroring backend data structures.
- Helps with hypermedia-driven APIs, where resources represent capabilities rather than static data.

Supports Evolutionary API Design

- If API models are decoupled from internal implementations, changes to one layer don't force changes to another.
- Makes APIs more resilient and future-proof.

Optimizes API Payloads for Real-World Usage

- Message models should be designed for actual client needs, rather than exposing internal data models wholesale.
- Leads to more efficient API interactions by sending only what's needed.

"Amundsen's Maxim" is a critical principle for designing flexible, resilient APIs. By keeping data, objects, resources, and messages distinct, API designers create systems that are more adaptable, evolvable, and client-friendly. This is especially relevant for affordance-driven designs, where API interactions are dynamic rather than rigidly predefined.

Selected Citations

These selected references illustrate that Amundsen's maxim is widely recognized and applied in discussions about effective API design, reinforcing the importance of distinguishing between various models to create flexible and maintainable APIs.

1. CodeOpinion Blog Post (2016): This article references Amundsen's tweet and discusses the pitfalls of directly exposing the data model through APIs, particularly in the context of OData with .NET and Entity Framework.

2. OpenTravel Tech FAQs: The OpenTravel Alliance references Amundsen's maxim in their technical FAQs to emphasize the distinctions between different models in API design.
3. APIscene Article (2024): An article on APIscene discusses "RESTful API Patterns and Practices," mentioning Amundsen's maxim in the context of data patterns and the importance of hiding storage internals from clients.
4. StackOverflow Discussion (2023): In a StackOverflow thread about structuring data objects in C# Web API, a user references Amundsen's maxim to highlight the importance of distinguishing between different models to avoid frontend issues.
5. Nordic APIs Article (2018): An article revisiting REST state machines cites Amundsen's maxim to argue against designing APIs as mere CRUD wrappers around databases, advocating for a more nuanced approach.

Mike Amundsen

Author, Speaker, Advisor

An internationally known author and speaker, Mike Amundsen consults with organizations around the world on network architecture, Web development, and the intersection of technology and society. He works with companies large and small to help them capitalize on the opportunities APIs, Microservices, and Digital Transformation present for both consumers and the enterprise.

Amundsen has authored numerous books and papers. His latest book is "Web API Patterns and Practices Cookbook" (2022). Amundsen's "Design and Build Great APIs" (2020) for Pragmatic Programmers Publishing is a popular developer-centric book. Amundsen also contributed to the O'Reilly Media book, "Continuous API Management" (2021,2018). His "RESTful Web Clients", was published by O'Reilly in February 2017 and he co-authored "Microservice Architecture" (June 2016). Amundsen's 2013 collaboration with Leonard Richardson "RESTful Web APIs" and his 2011 book, "Building Hypermedia APIs with HTML5 and Node", are common references for building adaptable Web applications.

Contacts

- Mastodon: https://mastodon.social/@mamund
- Github: http://github.com/mamund
- LinkedIn: http://linkedin.com/in/mamund
- YouTube: http://youtube.com/mamund
- Presentations: http://mamund.com/talks/
- Amazon Page: http://amazon.com/author/mamund