Headless & API date

Milano • 27 settembre 2019

API Design

I «must have» per governare un’Architettura Headless

Denis Signoretto - IT Architect & Senior Project Manager

I must have per passare ad un'Architettura Headless

27 Mar 2019 – Nessun commento – by Denis Signoretto

Nel precedente articolo "Architettura IT: il futuro è Headless" abbiamo introdotto alcuni concetti chiave delle **Architetture Headless**, evidenziando come la loro funzione sia finalizzata a **supportare strategie digitali evolute**. Nello specifico, un'infrastruttura IT headless supporta l'integrazione di interfacce utente moderne in modo indipendente dai sistemi di backend e dai relativi vincoli tecnologici, disponendo una gestione centralizzata di contenuti e servizi e garantendo **esperienze digitali coerenti e soddisfacenti agli utenti**.

Per passare ad un approccio headless **non basta solo la tecnologia, ma è indispensabile un approccio strategico.** Gli elementi abilitanti sono certamente le API ma per evolvere un'infrastruttura IT verso un paradigma headless, non è sufficiente esporre API REST: serve definire una roadmap che supporti questa evoluzione, affrontando anche le sfide che il mondo dell'API Management richiede.

6 elementi strategici per un approccio headless

1. La sicurezza dei dati *über alles*
Come punto di partenza fondamentale per passare ad un approccio strategico headless, è importante che l'infrastruttura identificata garantisca e offra servizi di:

- **Autenticazione delle API,** adottando standard che assicurino un'integrazione semplice con livelli di sicurezza

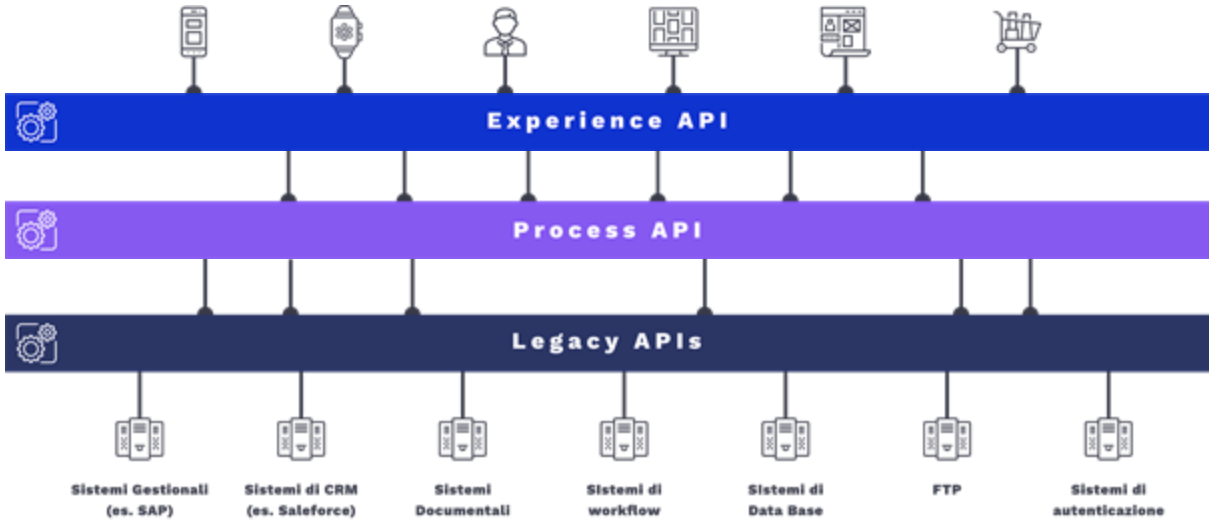
6. Analisi del dominio e design delle API

In ultima, ma non per questo di secondaria importanza, è fondamentale che le API siano strutturate secondo **un modello che rappresenti le entità del dominio, le proprie modalità di business,** le depuri e le renda indipendenti dalle complessità dei sistemi legacy creando un adeguato **strato di interfaccia** che favorisca la **riusabilità** e **isoli i client dalla complessità e dalle interdipendenze dei sistemi sottostanti**.

A step back, first ...

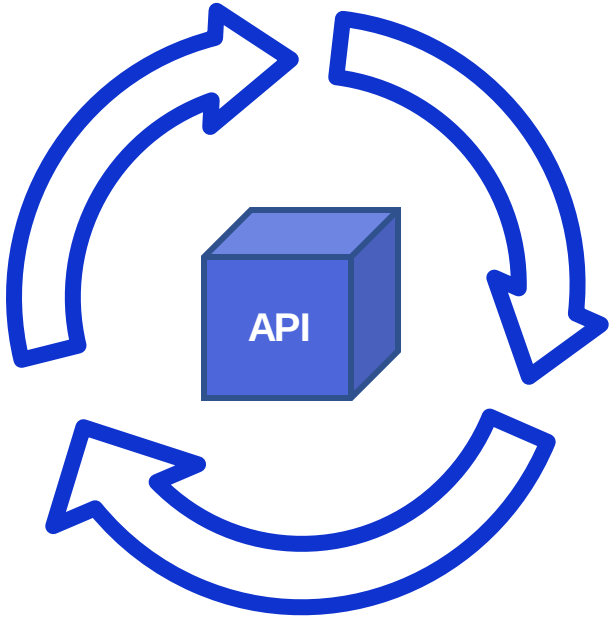
API vs Web Services

No siloes & Multiple Consumers



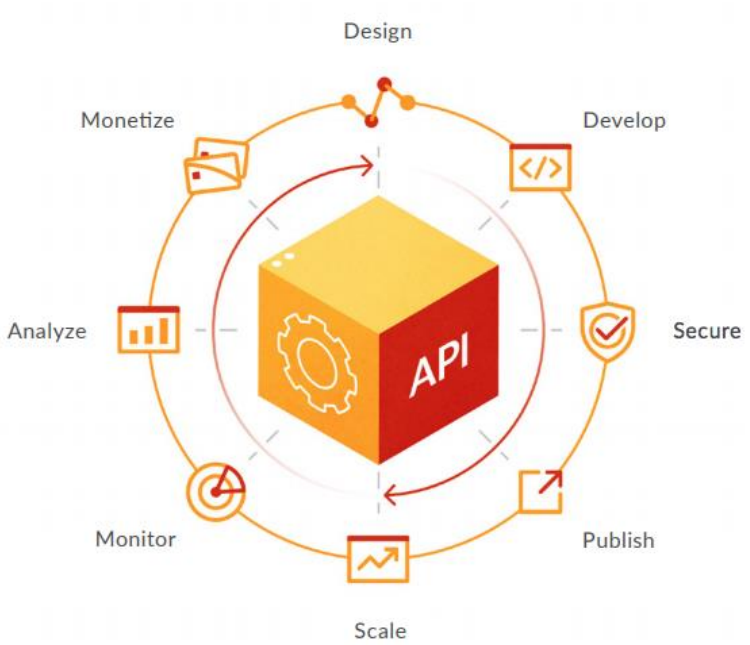
API vs Web Services

Reuse



API as a Product

Life-Cycle



API as a Product

SIDE A

Who are the API customers?

Developers

What are their needs?

$> Accessible and discoverable API

$> Documentation

$> Standards

$> Adoption of known best practices and conventions

$> Sandbox

$> Examples

$> Accessible and discoverable API

$> Documentation

$> Standards

$> Adoption of known best practices and conventions

$> Sandbox

$> Working Examples !!!

SIDE B

Is there an «API Product Manager»?

API Design
is an important pillar

How to put API Design into Practice ?!?

Case History Journey

in domain Modeling & Rest API Design

API Design approach

Domain Design

- Functional & Non-functional Requirements
- Summary of terms
- Use Case, Scenarios and Acceptance Criteria
- Flow and Sequence Diagrams

API Desing

- Consumer-Oriented Design
  - “Outside-in” approach
  - API First Design
- Agile Design
- Mock and Simulation

Domain & API

Specification Languages

Domain Modeling

UML Use Cases

Scenarios and acceptance criteria

As a logged-out user

I want to be able to sign in to a website

So that I can find acess my personal profile

Scenario: System user signs in with valid credentials

Given I'm a logged-out system user

and I'm on the Sign-In page

When I fill in the "Username" and "Password" fields with my authentication credentials

and I click the Sign-In button

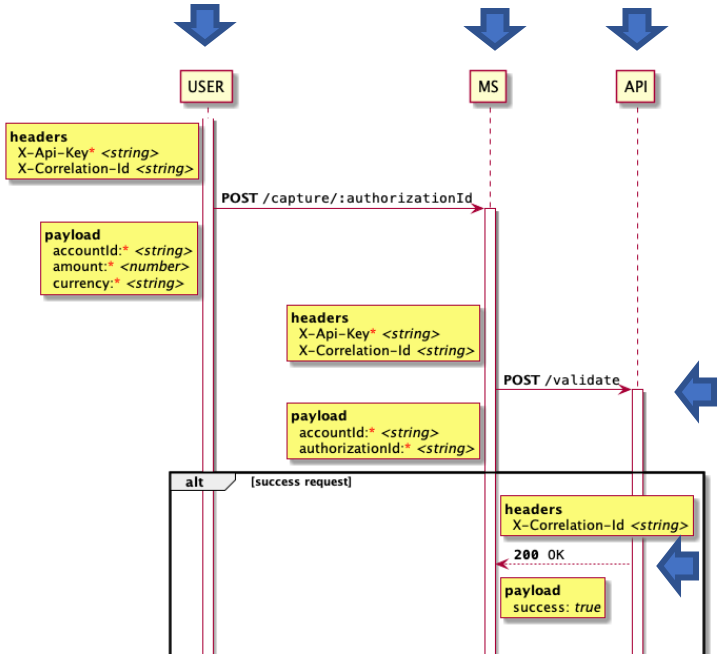
Then the system signs me in"

***Fonte** https://rubygarage.org/blog/clear-acceptance-criteria-and-why-its-important

UML Sequence Diagram

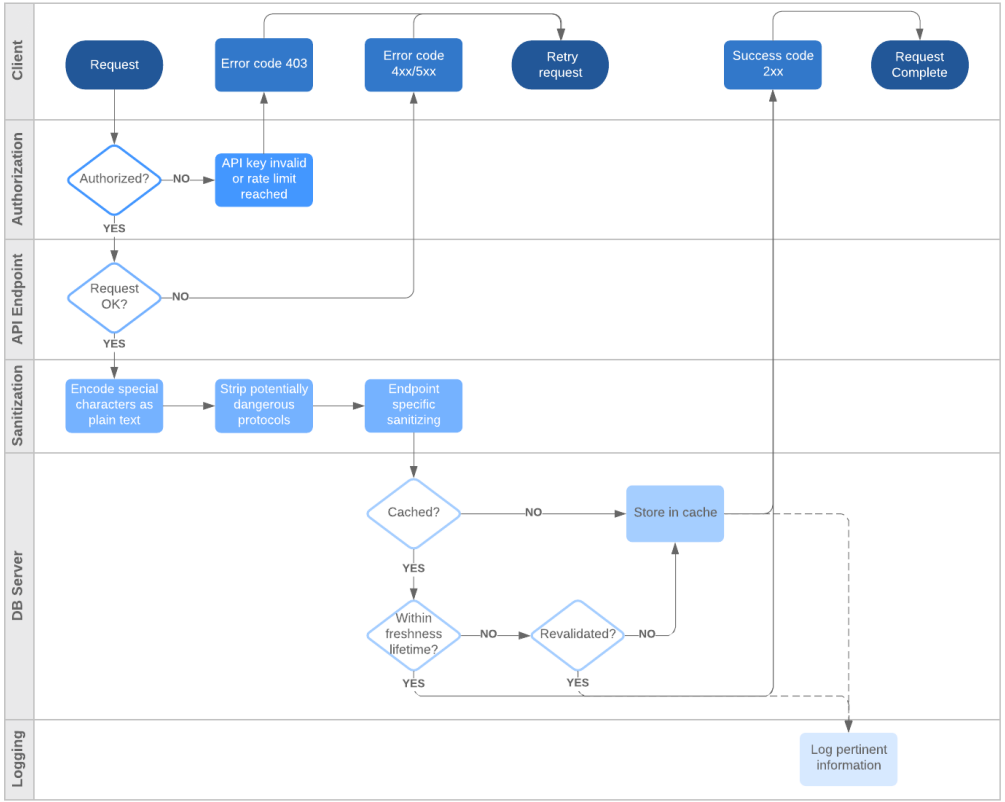
Sequence flow:

- Endpoints
- Resources Path & Parameters
- HTTP Verbs
- HTTP status codes
- Extra info: Headers



***Fonte** https://medium.com/@jccguimaraes/designing-an-api-6609eb771b18

UML Flow Diagram



Functional and non functional requirements

API Design

«Outside-in» vs «Inside-out»
and
API First approach

Rest API Design Language

OpenAPI (aka Swagger)

Rest API Definition

- Entities Schema
- Resources and paths
- HTTP Verbs and HTTP status codes
- Media Types
- Documentation (paths, entities, parameters,...)
- Examples
- ... and

:(

Your PC ran into a problem and needs to restart. We're just collecting some error info, and then we'll restart for you.

20% complete

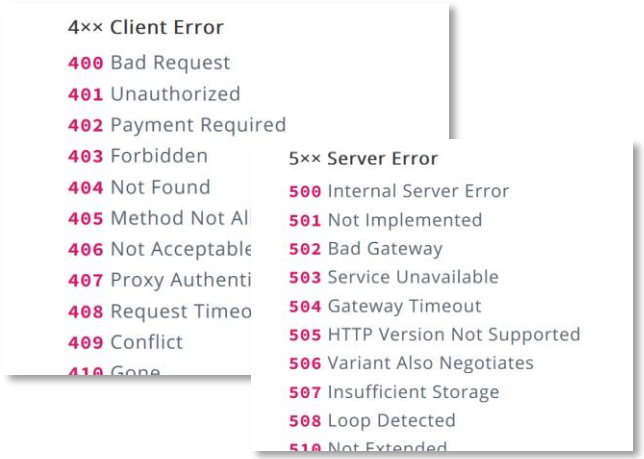
For more information about this issue and possible fixes, visit https://www.windows.com/stopcode

If you call a support person, give them this info:

Stop code: CRITICAL_PROCESS_DIED

Rest API Definition

HTTP STATUS CODES



ERRORS generated by intermediate Layers

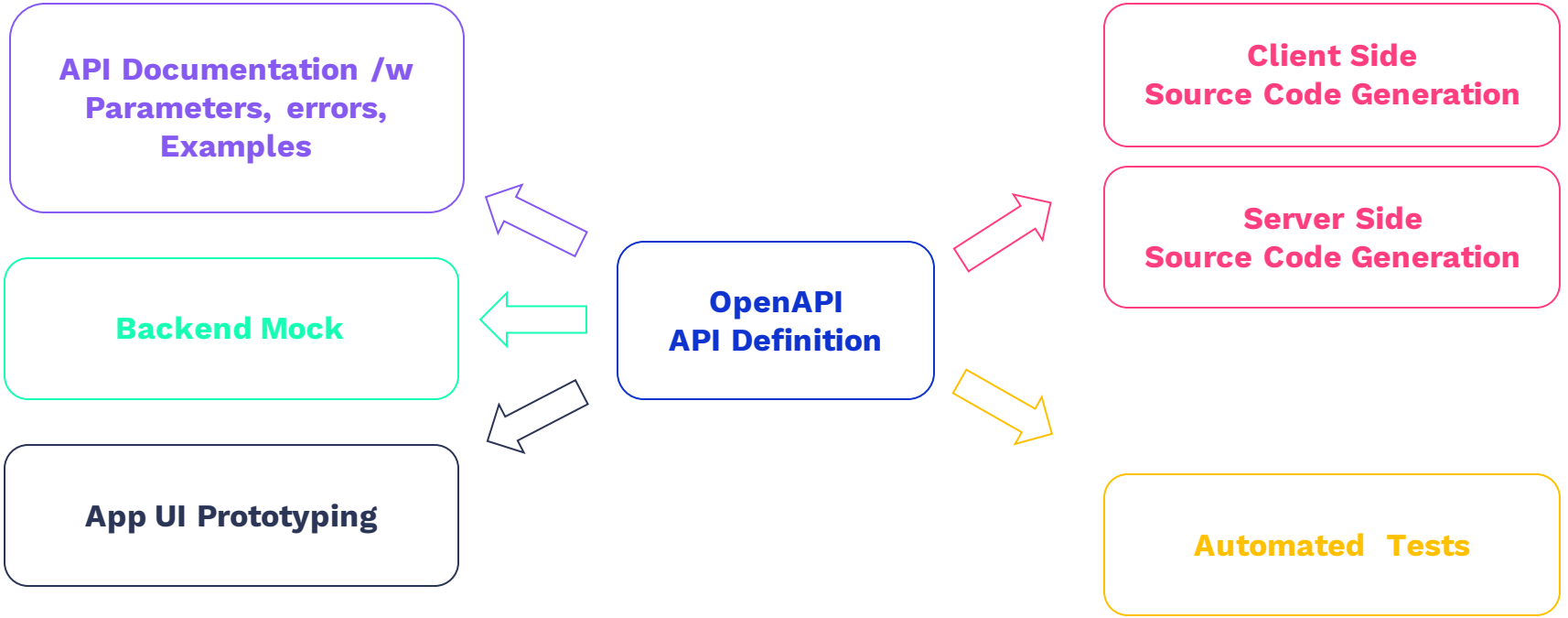
- Reverse Proxies
- API Gateway
- EBS
- …

DESIGN for problems

- **RFC 7807 (JSON Problem)**
- Google Errors, Facebook Errors,…

- Error code
- Human Error message
- Error description
- Internal reference ID & documetation

API First advantage



Conclusions

Importance of API Design

Un buon design rende **efficienti**
ed **efficaci** gli **sviluppatori** sia in fase
di prototipazione e integrazione
sia in fase di **problem solving** e **supporto**

Importance of API Design

Cambiare le API una volta pubblicate
impatta sul ciclo di vita delle API

API Architect

- Comprende i **bisogni operativi** degli sviluppatori
- Introduce, fa comprendere e utilizzare gli **strumenti di specifica**
- Assicura tecnicamente **qualità**, **documentazione**, **esempi**
- Garantisce **uniformità** e **adesione** agli **standard**
- Garantisce la **sicurezza**
- **Monitoring** ai fini delle **performance** e **disponibilità** delle API

API Product Manager

- Comprende i **bisogni degli utilizzatori in termini di funzionalità**
- Assicura strumenti per garantire **documentazione**, **facilità** di **utilizzo** e **integrazione**
- Promuove l'adozione delle API nel sistema azienda
- **Tiene sotto controllo e massimizza il ROI (Return of Investment)**
- Monitoring e **valuta KPI delle API**

Headless & API date

Milano • 27 settembre 2019

Grazie

Denis Signoretto
IT Architect & Senior Project Manager

@denissignoretto

Email: denis.signoretto@intesys.it