**REST API**Web service: “è un sistema software identificato da un URI ed ha delle interfacce pubbliche esposte con uno schema predefinito.

L’interazione avviene tramite messaggi tramite protocolli Internet.”

Il server risponde con la rappresentazione della risorsa.

REST non è uno standard

Rest si basa sui vincoli

Rest ha un interfaccia uniforme

Messaggi auto-descrittivi: ogni messaggio contiene informazioni sufficienti per descrivere l’elaborato.

HATEOAS: l’applicazione è pilotata da dei collegamenti ipertestuali.

REST è stateless:

ogni richiesta deve contenere tutte le informazioni necessarie per comprendere la richiesta senza contare su info di contesto nel server

REST è cachable:

Come nel World Wide Web, i clienti possono memorizzare nella cache le risposte

REST è client-server

REST è un sistema stratificato, ossia vi sono componenti intermedi che stanno tra il client e il server(proxy, load-balancer)

Server intermediari possono migliorare la scalabilità del sistema.

Gli strati possono anche implementare policy di sicurezza

Può supportare codice on demand

I server sono in grado di estendere o personalizzare la funzionalità di un client trasferendo logica in modo che esso la possa eseguire temporaneamente. Esempi di questo possono includere componenti compilati (applet Java, obsoleto) e script lato client come JavaScript

Se un API soddisfa tutti i vincoli è detto servizio RESTful

http access pattern

CRUD via http

Idea base è di avere una Resource, con un URI, una Representation, un insieme standard codificato di Operations

Immagine che contiene tavolo

Descrizione generata automaticamente

La maggior parte dei servizi web usa API REST

I web services possono essere documentati e descritti tramite:

WSDL (Web service Description Language)

WRDL (Web Resource Description Language)

Tipicamente un servizio consiste di documenti XML, di cui XML schemas ne danno una parziale descrizione Quello che non descrivono sono le transizioni da un documento all’altro cioè il “service's runtime behaviour” ß descritto da WRDL

WADL (Web Application Description Language)

**REST Design**

i metodi http hanno le proprietà di:

* Safe: semantica read-only, non si alterano le risorse sul server (GET deve essere safe)
* Idempotente: ripetendo una chiamata non si hanno dati collaterali (GET, PUT, DELETE)
* Cacheable: le risposte possono essere memorizzate per ri-usi futuri

Immagine che contiene tavolo

Descrizione generata automaticamente

linee guida per una buona progettazione di API REST

* **Nomi delle risorse e URI**
* **Documentazione delle API**
* **Versioni**
* **HTTP CRUD pattern**
* **Uso dei codici di risposta HTTP**
* **Parametri aggiuntivi in richieste**
* **Alcuni Patterns e Anti-patterns**

Risorsa specifica: oggetto o rappresentazione di entità significative nel dominio applicativo

Collezione: insieme di risorse omogenee

URI: è il path della risorsa con il quale si possono eseguire operazioni

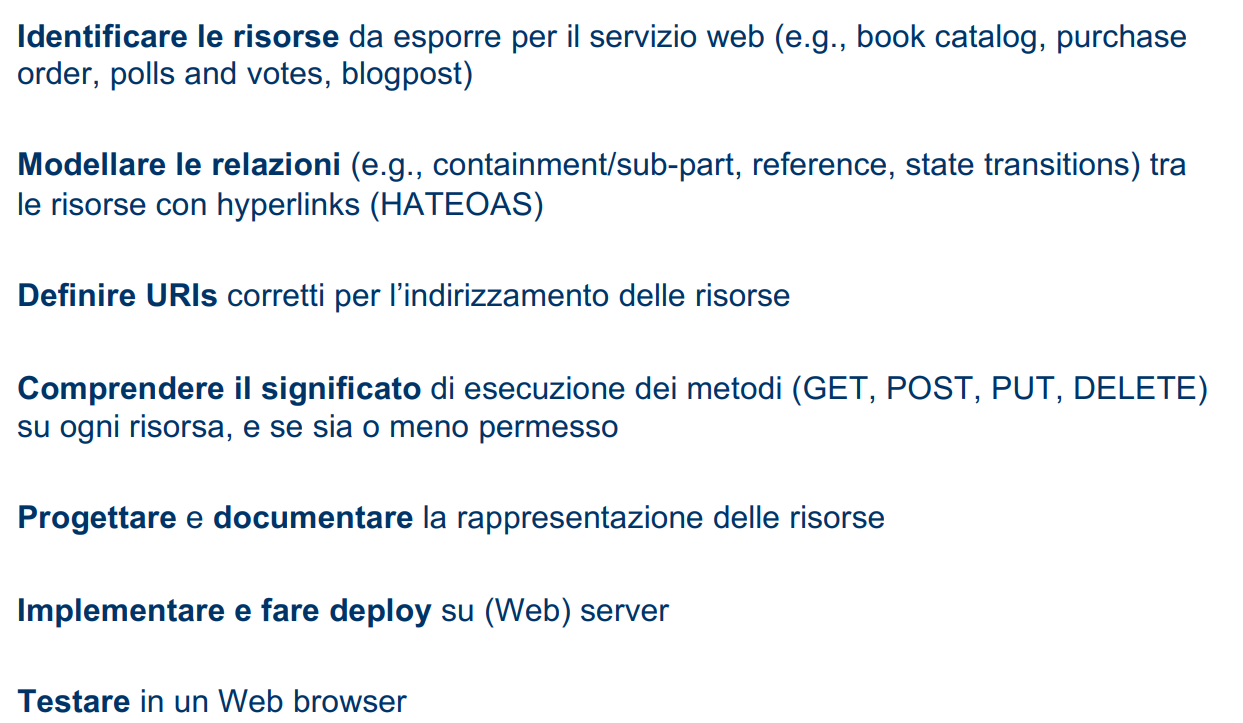
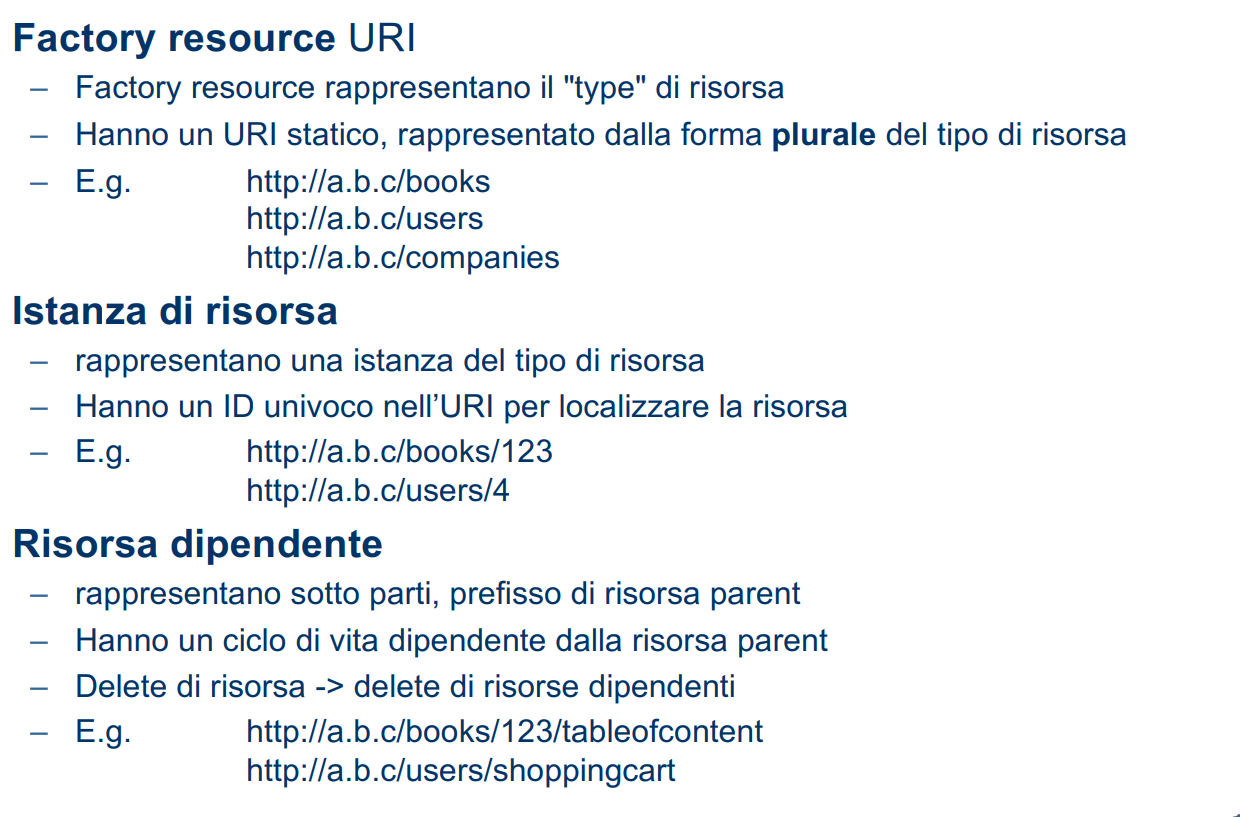


Immagine che contiene tavolo

Descrizione generata automaticamente

Ci sono diversi pattern per sviluppare REST



Tramite un suffisso si può definire il formato di rappresentazione (html,json…)