

API Manager APIMBBONE

Manuale per pubblicazione API

Versione	PARAGRAFO O PAGINA	DESCRIZIONE DELLA VARIAZIONE
V1.0.0	Tutto il documento	Versione iniziale del documento
V1.0.1	Capitolo 6.2	Modificato processo di richiesta pubblicazione API
V1.0.2	Capito 4, capitolo 6.1.2.1, Appendice C	Aggiunta descrizione tag per api di tipo shared ed exp
V2.0.0	Paragrafo 6.1	Aggiornamento per passaggio a WSO2 API Manager 4.1.0
V2.0.1	Capitolo 6.2 e sottosezioni	Modificate le indicazioni per apertura ticket
V2.0.2	Capitolo 4	Aggiornamento descrizione ambiente di test per esposizione su internet

Sommario

1	Scopo del documento.....	3
2	Riferimenti.....	3
3	API Management.....	3
3.1	Gestione della sicurezza	4
4	Istanze di piattaforma disponibili	4
5	Flusso di pubblicazione API su API Manager	5
6	Creazione e pubblicazione API	7
6.1	Documentazione richiesta per la pubblicazione API	7
6.1.1	Parametri di esposizione	7
6.2	Processo di richiesta pubblicazione API	12
6.2.1	Tipologie di richieste.....	12
6.2.2	Come inoltrare una richiesta	12
7	Rilasci nuova versioni di API esistenti.....	12
8	Appendice A - Elenco ambiti.....	14
9	Appendice B - Elenco enti.....	15
10	Appendice C - Elenco tags predefiniti.....	16
11	Appendice D - Gestione del traffico API	17
11.1	Throughput massimo del back-end	17
11.2	Livelli di sottoscrizione	17

1 Scopo del documento

Il documento ha come scopo quello di fornire le indicazioni su come pubblicare un API sulla piattaforma API Manager. APIMBBONE è stato progettato per gestire, tramite API, l'interazione e l'integrazione tra le PA, i cittadini e le imprese e per il supporto dell'iniziativa Backbone API, che ha l'obiettivo di promuovere la pubblicazione di API di utilità trasversale (Shared API) utili per ambiti vari e differenti dal proprio.

2 Riferimenti

- [1] Linee guida Modalità Interoperabilità via API [LG Arch APIInterop V1.1.0](#)
- [2] Linee guida progettazione API Rest [LG Arch-API-Rest-V03](#)
- [3] Manuale per Sottoscrizione API [LG APIMBBONE-APIStore Sottoscrizione V2](#)
- [4] Template per richiesta pubblicazione API [Template richiesta pubblicazione su Teknos](#)
- [5] Form per la creazione guidata della richiesta di pubblicazione <https://api-piemonte-pub.ecosis.csi.it/papis-request>

3 API Management

Le soluzioni di API Management costituiscono l'elemento abilitante per arricchire e personalizzare l'interazione tra le applicazioni che richiedono l'accesso ad API e i servizi di business che espongono le informazioni utili per la composizione delle applicazioni stesse. L'introduzione di un layer di astrazione tra i servizi che espongono funzionalità (API Providers) e le applicazioni che li consumano (API Consumers) semplificano gli sviluppi e favoriscono il disaccoppiamento tra i due livelli.

L'API Manager è la soluzione che permette di centralizzare il punto di ingresso per le chiamate, applicare politiche di throttling efficienti, monitorare le risorse utilizzate, tracciare le chiamate dei fruitori delle API di business ed ultimo ma non meno importante securizzare i servizi API.

Gli obiettivi principali dell'API Manager possono essere riassunti come segue:

- Gestione della sicurezza: garantisce l'accesso soltanto agli utenti/sistemi autorizzati ed evita l'uso improprio delle risorse protette. Sono disponibili diversi framework di sicurezza, lo scenario attuale implementato prevede l'adozione di OAuth2
- Gestione del traffico: performance gestite in maniera puntuale e dinamica per ciascuna API con limitazione del traffico in ingresso (rate limiting), applicazione di politiche di accesso diversificate in base sistema chiamante (throttling), routing e cache dei messaggi. Filtraggio del traffico in ottica di identificazione e neutralizzazione di minacce
- Gestione del ciclo di vita delle API: fasi di sviluppo, test, produzione e dismissione, nonché versionamento. Questa funzionalità garantisce la coerenza di differenti versioni dell'API consentendo il suo utilizzo da parte di diverse tipologie di utenti, in ambienti diversi e con diversi gradi di maturità
- Audit del sistema: tracciamento attività e statistiche di utilizzo.

Le componenti principali dell'API Manager sono le seguenti:

- **Publisher**, applicazione web che permette agli API Providers di creare e pubblicare le API. Nel Publisher si definiscono tutti gli aspetti di configurazione dell'API, compresi endpoint dei servizi di back-end e politiche di throttling e rate limiting.
- **API Store**, Store dedicato agli sviluppatori e alle terze parti che intendono integrare le API nelle loro applicazioni, il portale ospita la documentazione di supporto, monitorare l'utilizzo delle API nonché accedere ad altri strumenti di comunicazione e condivisione.
- **API Gateway**, componente di runtime dell'API Manager che ha come finalità essenziale di esporre i servizi messi a disposizione dall'intero sistema in maniera sicura, facilmente fruibile e controllata. L'API Gateway, dal punto di vista architetturale, è un proxy dei servizi esposti dai sistemi di back-end, in modo tale che tutti i sistemi fruitori debbano effettuare l'accesso a servizi e risorse attraverso questo componente. Dal punto di vista funzionale, il Gateway, riceve le richieste per accedere alle API ed attua le politiche di controllo di accessi, applica le regole di rate limiting e throttling e instrada le richieste verso i sistemi di back-end.
- **Key Manager**, componente che ha il compito di gestire tutte le questioni relative alla sicurezza e alle chiavi. Tutte le richieste di generazione di nuovi access token sono gestite da questo componente che effettua la validazione di tutti i parametri inviati nella richiesta (client_id, client_secret, username, password, ecc...).
- **Traffic Manager**, componente che si occupa di regolare il traffico di ciascuna API secondo le politiche che sono state definite in fase di definizione dell'API e in fase di sottoscrizione alla stessa. Il motore di elaborazione del Traffic Manager elabora le politiche di throttling in real time, incluse le politiche di rate limiting delle chiamate alle API.

3.1 Gestione della sicurezza

Open Authorization 2 (OAuth2) è lo standard de facto supportato dall'API Manager per l'autorizzazione a risorse (API) protette da autenticazione senza necessità di dover condividere pubblicamente tali credenziali. Il protocollo OAuth permette di autorizzare una terza parte a gestire sezioni riservate di una risorsa senza vincolare il sistema a mettere in circolazione le credenziali per accedervi. I vantaggi offerti da questa tecnologia derivano direttamente dalla possibilità di avere un accesso in autenticazione mediante generazione di un token di autorizzazione apposito.

Per la generazione degli access token si possono utilizzare diversi grant type OAuth 2.0 in base ai casi d'uso applicativi, di seguito i grant type supportati:

- client credentials
- authorization code
- refresh token
- resource owner password

Per le specifiche dei grant type si rimanda alle linee guida "Modalità Interoperabilità via API" [\[1\]](#) e al sito ufficiale della OAuth 2.0 industry-standard protocol for authorization <https://oauth.net/2/grant-types/>.

4 Istanze di piattaforma disponibili

La piattaforma API Manager APIMBBONE dispone delle seguenti filiere:

- **istanza di test:** esposta sulla rete di test, dedicata alla fruizione di servizi di test. Per la sola parte di runtime (gateway), è esposta sia sulla rete interna CSI sia su internet. La parte di gestione delle applicazioni e sottoscrizioni (API Store) è esposta solamente su rete interna CSI e può essere fatta solamente da utenti CSI. Questa filiera può essere utilizzata per l'esposizione di API per permettere i test di integrazione.
- **istanza di produzione:** esposta sulla rete di produzione, disponibilità di esposizione internet e rete interna CSI per esposizione delle API di produzione. È inoltre disponibile in produzione l'ambiente sandbox che permette di disporre di una API con doppio puntamento, nello specifico l'esposizione sandbox abilita scenari di test/collauda per fruitori esterni esposti su rete internet.

5 Flusso di pubblicazione API su API Manager

Per descrivere il flusso di pubblicazione di API su API Manager è necessario considerare le istanze dell'API Manager, gli stati del ciclo di vita delle API e gli attori coinvolti nel processo.

La matrice RACI di seguito descrive ruoli e responsabilità di ogni attore coinvolto nel ciclo di vita dell'API.

Gli attori evidenziati nella RACI sono i seguenti:

- Project Manager API Provider: coordinatore del progetto fornitore di API
- Progettista/Sviluppatore API Provider: progettista/sviluppatore del progetto fornitore di API
- Project Manager API Consumer: coordinatore del progetto che fruisce delle API esposte su API Manager
- Progettista/Sviluppatore API Consumer: progettista/sviluppatore del progetto che fruisce delle API esposte su API Manager
- API Platform Team: team amministratore delle istanze API Manager e della governance del catalogo API

Matrice RACI: Governo ciclo di vita API		Attori				
		API Provider		API Consumer		APIM Platform team
		API Project Manager	Progettista/Sviluppatore API	API Project Manager	Progettista/Sviluppatore API	
Attività ciclo di vita API						
Sviluppo	Progettazione API	A	R			C
	Sviluppo API	A	R			
API Manager: istanza di test	Creazione e pubblicazione	A	C			R
	Prototipazione	A	C			R
	Deprecazione	A	C			R
	Ritiro	A	C			R
	Blocco/sblocco	A	C			R
	Sottoscrizione	C		A	R	I
API Manager: istanza di produzione	Creazione e pubblicazione	A	C			R
	Prototipazione	A	C			R
	Deprecazione	A	C			R
	Ritiro	A	C			R
	Blocco/sblocco	A	C			R
	Sottoscrizione	C		A	R	I
R	Responsible	esegue ed assegna l'attività				
A	Accountable	responsabilità sul risultato dell'attività				
C	Consulted	collabora con il Responsible per l'esecuzione dell'attività				
I	Informed	informato al momento dell'esecuzione dell'attività				

Di seguito le fasi principali del ciclo di vita delle API con le relative responsabilità:

- **Progettazione API:** fase di design dell'interfaccia. La progettazione deve seguire l'approccio **contract first** rispettando le linee guida di progettazione Api Rest [2]. In questa fase gli attori coinvolti sono il Progettista e lo sviluppatore provider dell'API come esecutori dell'attività, il Project Manager provider dell'API come responsabile del risultato dell'attività. *API Platform Team è consultabile per un supporto in fase di definizione delle interfacce API.*
- **Sviluppo API:** fase di implementazione vera e propria dell'API. Gli attori coinvolti sono il Progettista e lo sviluppatore provider dell'API come esecutori dell'attività, il Project Manager provider dell'API come responsabile del risultato dell'attività.
- **Creazione e pubblicazione API:** fase in cui l'API viene resa visibile nello Store. Gli attori coinvolti sono il gruppo di amministrazione dell'API Manager come esecutore dell'attività, il Progettista e lo sviluppatore provider dell'API consultabile al bisogno, il Project Manager provider dell'API come responsabile del risultato dell'attività. *API Platform Team in fase di pubblicazione è responsabile di verificare l'aderenza alle linee guida di progettazione per le API Rest di tipo shared.*
- **Prototipazione:** stato opzionale dell'API sullo Store dell'API Manager. Lo stato "prototipo" di un API consiste in un mock-up dell'API, essa è accessibile pubblicamente senza bisogno di una sottoscrizione. Lo stato prototipazione è utile nel caso in cui si debba rendere disponibile ai fruitori in tempi rapidi l'interfaccia dell'API prima della sua implementazione. Gli attori coinvolti sono il gruppo di amministrazione dell'API Manager come esecutore dell'attività, il Progettista e lo sviluppatore provider dell'API consultabile per chiarimenti, il Project Manager provider dell'API come responsabile del risultato dell'attività.

- **Deprecazione:** stato opzionale dell'API sullo Store dell'API Manager. Lo stato "deprecato" di un API non permette nuove sottoscrizioni ma garantisce le precedenti sottoscrizioni da parte dei fruitori esistenti. Gli attori coinvolti sono il gruppo di amministrazione dell'API Manager come esecutore dell'attività, il Progettista e lo sviluppatore provider dell'API consultabile al bisogno, il Project Manager provider dell'API come responsabile del risultato dell'attività.
- **Ritiro:** stato opzionale dell'API sullo Store dell'API Manager. Lo stato "ritirato" di un API comporta la cancellazione dell'API dallo Store. Gli attori coinvolti sono il gruppo di amministrazione dell'API Manager come esecutore dell'attività, il Progettista e lo sviluppatore provider dell'API consultabile in caso di necessità, il Project Manager provider dell'API come responsabile del risultato dell'attività.
- **Blocco/Sblocco:** stato opzionale dell'API sullo Store dell'API Manager. Lo stato "blocco/sblocco" di un API comporta un'indisponibilità temporanea dell'API dallo Store. Gli attori coinvolti sono il gruppo di amministrazione dell'API Manager come esecutore dell'attività, il Progettista e lo sviluppatore provider dell'API consultabile al bisogno, il Project Manager provider dell'API come responsabile del risultato dell'attività.
- **Sottoscrizione:** fase che identifica il contratto tra l'API Consumer e API Provider. Gli attori coinvolti sono il Progettista e lo sviluppatore dell'API Consumer come esecutori dell'attività, il Project Manager API Consumer come responsabile del risultato dell'attività. Il gruppo di amministrazione dell'API Manager solo informato e l'API Provider Project Manager consultabile al bisogno. Le modalità operative della sottoscrizione API sono descritte nel manuale per Sottoscrizione API [\[3\]](#)

6 Creazione e pubblicazione API

Per poter pubblicare un API è necessario partire da una documentazione a corredo della richiesta di pubblicazione, che permetta all'amministratore di censirla nell'API Manager in autonomia senza dover contattare il referente di back-end (o almeno di limitare le richieste di informazioni).

6.1 Documentazione richiesta per la pubblicazione API

Per i servizi di tipo **REST** la documentazione richiesta è composta da:

- definizione dell'API: è la definizione OAS (OpenAPI Specification in versione 2 o 3) utilizzata per effettuare il design dell'API; può essere fornita come file, in formato JSON o YAML, oppure può essere fornita la URL del file, accessibile da rete Nivola e non protetta da autenticazione
- parametri di esposizione: sono i parametri di esposizione dell'API sull'istanza API Manager; consiste in un file in formato JSON (descritto in dettaglio nel paragrafo [Parametri di esposizione](#))

Per i servizi di tipo **SOAP** la documentazione richiesta è composta da:

- wsdl del servizio: definizione del servizio SOAP esposto; può essere fornita come file oppure può essere fornita la URL, accessibile da rete Nivola e non protetta da autenticazione
- parametri di esposizione: sono i parametri di esposizione dell'API sull'istanza API Manager; consiste in un file in formato JSON (descritto in dettaglio nel paragrafo [Parametri di esposizione](#))

6.1.1 Parametri di esposizione

Consiste in un file, in formato JSON, con le configurazioni necessarie all'API Manager per poter pubblicare correttamente l'API.

Per la compilazione del file JSON ci sono due possibilità:

- scaricare il template della richiesta di pubblicazione [\[4\]](#) e compilarlo seguendo le indicazioni descritte nel paragrafo successivo.
- utilizzare la form online [\[5\]](#), che genera il file JSON

6.1.1.1 Struttura del file JSON per la richiesta di pubblicazione di una API

La struttura e gli attributi del file JSON sono descritti di seguito.

```
{
  "name": "my-service-soap",
  "description": "This document describe a RESTful API for testing purpose.\r\n",
  "context": "/my-service-soap",
  "version": "v1",
  "provider": "admin",
  "responseCachingEnabled": false,
  "cacheTimeout": 300,
  "type": "SOAP",
  "transport": [
    "http",
    "https"
  ],
  "tags": [
    "testsuite",
    "test"
  ],
  "policies": [
    "Unlimited"
  ],
  "maxTps": {
    "sandbox": 500,
    "production": 100
  },
  "apiThrottlingPolicy": "Unlimited",
  "endpointConfig": {
    "production_endpoints": {
      "url": "https://localhost:9443/services/Version/"
    },
    "sandbox_endpoints": {
      "url": "https://localhost:9443/services/Version/"
    },
    "endpoint_type": "http",
    "endpoint_security": {
      "sandbox": {
        "password": "1234",
        "tokenUrl": null,
        "clientId": null,
        "clientSecret": null,
        "customParameters": null,
        "additionalProperties": {},
        "type": "BASIC",
        "grantType": null,
        "enabled": true,
        "uniqueIdentifier": null,
        "username": "TEST"
      },
      "production": {
        "password": "1234",
        "tokenUrl": null,
        "clientId": null,
        "clientSecret": null,
        "customParameters": null,
        "additionalProperties": {},
        "type": "BASIC",
        "grantType": null,
        "enabled": true,
        "uniqueIdentifier": null,
        "username": "TESTPROD"
      }
    }
  },
  "businessInformation": {
    "businessOwnerEmail": "apimint@csi.it",
```



```

    "technicalOwnerEmail": "apimint@csi.it",
    "technicalOwner": "Samwell Tarly",
    "businessOwner": "John Snow"
  },
  "corsConfiguration": {
    "accessControlAllowOrigins": [
      "*"
    ],
    "accessControlAllowHeaders": [
      "authorization",
      "Access-Control-Allow-Origin",
      "Content-Type",
      "SOAPAction"
    ],
    "accessControlAllowMethods": [
      "GET",
      "PUT",
      "POST",
      "DELETE",
      "PATCH",
      "OPTIONS"
    ],
    "accessControlAllowCredentials": false,
    "corsConfigurationEnabled": false
  },
  "additionalProperties": [
    {
      "name": "institution",
      "value": "CSI",
      "display": true
    }
  ],
  "categories": [],
  "gatewayVendor": "wso2",
  "gatewayType": "wso2/synapse"
}

```

Descrizione delle informazioni richieste:

- **name (stringa):** il nome dell'API visibile nell'API Store anche per gli utenti non loggati, trattandosi di una label di business deve essere valorizzata tenendo in considerazione la finalità dell'API. È consigliato evitare nomi puramente tecnici ma usare nomi parlanti (es. toponomastica-comune-torino, catasto-impianti-termici-rp). Il nome non può contenere spazi; gli unici caratteri non alfanumerici ammessi sono "-" (trattino), "_" (underscore) e "." (punto). Massimo 50 caratteri
- **description (stringa):** la descrizione dell'API, deve essere una descrizione sintetica di business dell'API.
- **context (stringa):** il contesto con cui verrà esposta sull'API Manager, deve rispettare il formato `/API_GROUP_PATH/API_ID` (per il dettaglio su come definire il contesto si rimanda al documento LG_Arch_APIInterop_V1.0.0) dove:
 - API_GROUP_PATH: ambito di appartenenza dell'API, deve essere uno di quelli presenti nell'[Appendice A - Elenco ambiti](#) del presente documento
 - API_ID: codice univoco identificativo dell'istanza di API
- **version (stringa):** la versione dell'API, deve rispettare il seguente formato `v<numero>` (es. v1, v2, ecc...).
- **responseCaching (stringa):** abilitazione della cache della response a livello di API Manager, per migliorare le performance non sollecitando il back-end in caso di molteplici chiamate con i medesimi dati. Se abilitato, settare un corretto timeout nel campo *cacheTimeout*. Possibili valori sono: Enabled o Disabled. Il default è Disabled.
- **cacheTimeout (numero):** timeout della cache della response. Il default è 300 secondi.

- **type (enum):** "HTTP" per le API di tipo REST, "SOAP" per le API di tipo SOAP
- **transport (array di enum):** protocollo di esposizione dell'API, "http" e/o "https".
- **tags (array di stringhe):** parole chiave che permettano di classificare l'API al fine di agevolarne la ricerca. Inserendo il tag **<nome gruppo>-group** è possibile organizzare le API in cartelle, <nome gruppo> solitamente corrisponde all'ambito dell'API. I tag devono essere scritti in down case senza spazi o caratteri speciali. Le api di tipo shared o di esposizione devono essere taggate con i tag predefiniti **shared** ed **exp**. Devono essere presenti almeno i seguenti tags:
 - **<Nome del gruppo>-group:** per il raggruppamento, in visualizzazione sull'API Store, delle API di un ambito
 - **<ambito>:** l'ambito dell'API; deve essere uno degli ambiti presenti nell'elenco presente nell'Appendice A del presente documento
 - **<ente>:** il nome dell'ente scritto per esteso (es. regione piemonte, cittadino romano). Fare riferimento all'Appendice B per dettagli sulla convenzione di scrittura tag relativi agli enti.
 - **"cittadino" o "funzionario" o "impresa":** in base all'audience dell'API
 - **"a2a" o "a2c" o "a2b":** in funzione dell'interoperabilità erogata:
 - tra pubbliche amministrazioni
 - pubblica Amministrazione che eroga servizi a cittadini
 - pubblica Amministrazione che eroga servizi ad imprese
- **policies (array di enum):** livelli di sottoscrizione messi a disposizione per l'API (vedi [Appendice D](#)). I possibili valori sono:
 - **Unlimited**
 - **Gold** (5000 richieste per minuto)
 - **Silver** (2000 richieste per minuto)
 - **Bronze** (1000 richieste per minuto)
 - **10PerMin** (10 richieste per minuto)
- **maxTps (oggetto):** numero di transazioni supportate dal backend (vedi [Appendice D](#))
 - **sandbox (numero):** numero di transazioni per secondo supportate dal back-end di sandbox
 - **production (numero):** numero di transazioni per secondo supportate dal back-end di produzione
- **apiThrottlingPolicy (enum):** definizione del throttling a livello di API, deve essere impostato a "Unlimited".
- **endpointConfig (oggetto):**
 - **production_endpoints (oggetto):**
 - **url (stringa):** endpoint del back-end di produzione
 - **sandbox_endpoints (oggetto):**
 - **url (stringa):** endpoint del back-end di sandbox
 - **endpoint_type (enum):** il tipo di endpoint; dovrà essere valorizzato con "http" nel caso di API Rest oppure "address" per API Soap.
 - **endpointSecurity (oggetto):** credenziali di autenticazione sul back-end, nel caso il back-end non abbia autenticazione quest'oggetto si può omettere
 - **sandbox (oggetto)**
 - **enabled (booleano):** true se l'endpoint di back-end è attiva un qualche tipo di autenticazione, false altrimenti
 - **type (enum):** tipo di security implementata sul back-end. I tipi supportati sono "BASIC" per la Basic Authentication, "DIGEST" per Digest Authentication e "OAUTH" se il back-end supporta protocollo OAuth2

- **username (stringa):** nel caso di type basic e digest, utente con il quale autenticarsi sul back-end; nel caso di type oauth e grant type password, è l'utente per il quale generare l'access token
- **password (stringa):** nel caso di type basic e digest, utente con la quale autenticarsi sul back-end; nel caso di type oauth e grant type password, è la password dell'utente per il quale generare l'access token
- **grantType (enum):** nel caso di type oauth, è il grant type con il quale generare l'access token; i possibili valori sono "PASSWORD" o "CLIENT_CREDENTIALS"
- **tokenUrl (stringa):** nel caso di type oauth, è la url per generare l'access token
- **clientId (stringa):** nel caso di type oauth, è il client id dell'applicazione creata sull'autorization server
- **clientSecret (stringa):** nel caso di type oauth, è il client secret dell'applicazione creata sull'autorization server
- **customParameters ():**
- **additionalProperties ():**
 - **production (oggetto)**
 - ...
- **businessInformation (oggetto):** informazioni di contatto
 - **businessOwner (stringa):** nome del referente di business
 - **businessOwnerEmail (stringa):** email del referente di business
 - **technicalOwner (stringa):** nome del referente tecnico
 - **technicalOwnerEmail (stringa):** email del referente tecnico
- **corsConfiguration:** configurazione per il CORS (Cross-Origin Resource Sharing)
 - **corsConfigurationEnabled (boolean):** impostare a *true* per abilitare specifiche configurazioni relative al CORS, *false* altrimenti
 - **accessControlAllowCredentials (boolean):** indica se dare accesso alle credenziali al codice del frontend
 - **accessControlAllowOrigins (array di stringhe):** indica le sorgenti con le quali condividere la risposta
 - **accessControlAllowHeaders (array di stringhe):** indica quali header possono essere usati nella richiesta
 - **accessControlAllowMethods (array di stringhe):** specifica quali metodi sono abilitati ad accedere ad una risorsa
- **additionalProperties (array di oggetti):** contiene le proprietà aggiuntive dell'API, si possono aggiungere tutte le proprietà necessarie e che saranno visibili sull'API Store nella sezione Metadati API. Le proprietà obbligatorie sono: **institution** (ente a cui appartiene l'API, es. "institution": "Regione Piemonte") e **api_product_id** (l'identificativo del prodotto dell'API censito sulla procedura Anagrafica Prodotti, strutturato come <cod prodotto>-<cod componente>-<linea cliente>, es. "api_product_id": "nao-orchanpr-coto-01")
 - **name (stringa):** nome della proprietà (es. "institution")
 - **value (stringa):** valore della proprietà (es. "Regione Piemonte")
 - **display (boolean):** se true la proprietà verrà mostrata nel catalogo delle API, false altrimenti
- **categories (array di stringhe):** le categories dell'API, i possibili valori sono "exp" e/o "shared",

- **gatewayVendor (enum):** l'unico valore ammesso è "wso2",
- **gatewayType (enum):** l'unico valore ammesso è "wso2/synapse"

6.2 Processo di richiesta pubblicazione API

Il progetto che necessita di pubblicare API su API Manager deve inoltrare la richiesta utilizzando lo strumento **Self Portal** disponibile al seguente link:

<https://selfportal-itsm.csi.it/>

6.2.1 Tipologie di richieste

Sono attualmente previste 4 macro-tipologie di attività:

Nuova API

Si tratta di tutte quelle attività che prevedono la configurazione di una nuova API sia in Fruizione che in Erogazione.

Variazione API

Tutte le attività di aggiornamento di una API esistente.

Dismissione API

La richiesta di dismissione di una API esistente

Variazione soggetto

La richiesta di variazione del soggetto intestatario di una API esistente

6.2.2 Come inoltrare una richiesta

Per sottomettere una richiesta di implementazione si dovrà accedere al portale Self Help, quindi selezionare "Richiesta di supporto" e navigare fino alla voce

Nome Piattaforma di interoperabilità API Manager

Tipologia Self Help

Categoria Self Help - Requester CSI Piemonte

Sottocategoria Infrastrutture - Piattaforma di interoperabilità API Manager


Una volta aperto il form di sottomissione richiesta, andrà selezionata come "Sottocategoria" la tipologia di richiesta desiderata e come "Servizio" il contesto a cui si riferisce.

Se la richiesta riguarda una nuova API o una variazione, dovranno essere allegati anche tutti gli eventuali files a corredo.

7 Rilasci nuove versioni di API esistenti

Il versionamento delle API è normato alle linee guida progettazione API Rest [\[2\]](#), di seguito alcuni passi fondamentali per un corretto versionamento.

Le API vengono identificate con una versione nel formato **v<numero>** e sono create a partire dalla versione v1, nelle versioni successive, l'incremento rappresenta una versione non retrocompatibile.

	<p align="center">API Manager APIMBBONE</p> <p align="center">Pubblicazione API su API Store</p>	<p align="center">LG_APIMBBONE- APIStore_Pubblicazion e_V2.0.2</p> <p align="center">Pag. 13 di 18</p>
--	--	---

Ogni API viene richiamata indicando esplicitamente la versione e **non sono usate versioni di default**.

In particolare, la versione dell'API dipende da quale evoluzione ha avuto il servizio di back-end corrispondente esposto dall'API Manager.

Di seguito le casistiche che si possono verificare:

- patch al servizio senza modifica dell'interfaccia. Qualora si riporti una variazione sul servizio senza modificarne l'interfaccia (ad esempio per la correzione di un'anomalia), non occorre effettuare alcuna modifica sull'API Manager. L'API mantiene la versione definita e tutti i fruitori usufruiranno della correzione.
- modifiche retrocompatibili al servizio. Qualora si riporti una variazione sul servizio che comporta modifiche dell'interfaccia, mantenendo la retrocompatibilità con la versione precedente, occorre:
 - installare la nuova versione del servizio di back end sovrascrivendo la precedente;
 - non occorre effettuare alcuna modifica sull'API Manager. L'API mantiene la versione definita e tutti i fruitori usufruiranno della correzione. Sono da considerarsi retrocompatibili solo le modifiche che aggiungono nuove operazioni su vecchi o nuovi tipi.
- modifiche non retrocompatibili al servizio:
 - installare la nuova versione del servizio di back end;
 - pubblicare la nuova versione dell'API (che renderà automaticamente obsoleta quella precedente diventando deprecata). In questo modo i sottoscrittori utilizzeranno due servizi di back end che dovranno coesistere il tempo necessario per permettere ai sottoscrittori la migrazione alla nuova API.

8 Appendice A - Elenco ambiti

Per completezza di riportano gli ambiti disponibili alle linee guida “Modalità Interoperabilità via API” [\[1\]](#):

- agricoltura: api attinenti alle tematiche relative al mondo dell’agricoltura
- ambiente: api attinenti alle tematiche relative al mondo dell’ambiente
- attproduttive: api attinenti alle tematiche relative al mondo delle attività produttive
- ammcontabile: api attinenti alle tematiche relative al mondo dell’amministrazione e contabilità
- gis: api cartografiche
- catasto: api attinenti alle tematiche relative al mondo del catasto
- cultura: api attinenti alle tematiche relative al mondo della cultura
- demografia: api attinenti alle tematiche relative al mondo della demografia
- documentale: api attinenti alle tematiche relative al mondo della gestione documentale
- modulistica: api attinenti alle tematiche relative al mondo della modulistica digitale
- edilizia: api attinenti alle tematiche relative al mondo dell’edilizia
- energia: api attinenti alle tematiche relative al mondo dell’energia
- fiscalita: api attinenti alle tematiche relative al mondo della fiscalità e dei tributi
- formazione: api attinenti alle tematiche relative alla formazione (non professionale)
- identita: api attinenti alle tematiche di identità digitale (autenticazione, autorizzazione, ...)
- lavoro: api attinenti alle tematiche relative al mondo del lavoro
- formazprof: api attinenti alle tematiche relative al mondo della formazione professionale
- istruzione: api attinenti alle tematiche relative al mondo dell’istruzione
- sanita: api attinenti alle tematiche relative al mondo della sanità
- patrimonio: api attinenti alle tematiche relative al mondo della gestione del patrimonio
- hr: api attinenti alle tematiche di gestione del personale (human resources)
- tecno: api a carattere tecnico, non tematiche
- sicurezza: api attinenti alle tematiche relative al mondo della sicurezza pubblica
- socioassist: api attinenti alle tematiche relative al mondo socio-assistenziale
- territorio: api attinenti alle tematiche relative al mondo del territorio
- trasporti: api attinenti alle tematiche relative al mondo dei trasporti
- turismo: api attinenti alle tematiche relative al mondo del turismo
- urbanistica: api attinenti alle tematiche relative al mondo dell’urbanistica
- logistica: api attinenti alle tematiche relative al mondo della logistica
- approvvig: api attinenti alle tematiche relative al mondo degli approvvigionamenti
- pagamenti: api attinenti alle tematiche relative al mondo dei pagamenti elettronici
- sport: api attinenti alle tematiche relative al mondo dello sport
- lavpubb: api attinenti alle tematiche relative al mondo dei lavori pubblici

9 Appendice B - Elenco enti

Di seguito l'elenco dei tag dei principali enti piemontesi:

Regione Piemonte → regionepiemonte

Regione Sanità → regpsanita

Città Metropolitana di Torino → cmtorino

Città di Torino → comunetorino

Di seguito la linea guida per la costruzione dei tag per enti locali o altri enti:

- “Prefisso” che identifica la tipologia di ente
- “Suffisso” che identifica univocamente il nome dell'ente omettendo spazi, preposizioni e accenti

Il tag si compone concatenando “Prefisso” e “Suffisso”.

Nomenclatura “Prefisso”:

Città Metropolitana → cm

Comune → comune

Regione → regione

ASL → asl

Alcuni esempi di enti locali o altri enti:

Comune di Pinasca → comunepinasca

ASL To1 → aslto1

Comune di Rivalta di Torino → comunerivaltatorino

Città Metropolitana di Roma → cmroma

Città di Milano → comunemilano

10 Appendice C - Elenco tags predefiniti

Di seguito l'elenco dei tags predefiniti obbligatori:

- regionepiemonte → Regione Piemonte
- regpsanita → Regione Sanità
- cmtorino → Città Metropolitana di Torino
- csi → CSI
- cittadino → Cittadino
- funzionario → Funzionario
- impresa → Impresa
- a2a → tra pubbliche amministrazioni
- a2c → pubblica Amministrazione che eroga servizi a cittadini
- a2b → pubblica Amministrazione che eroga servizi ad imprese

Di seguito l'elenco dei tags predefiniti opzionali:

- shared → api di tipo shared
- exp → api di esposizione per i test di integrazione

11 Appendice D - Gestione del traffico API

La componente di Traffic Manager permette di regolare il traffico di accesso alle API differenziando i livelli di servizio tra diversi fruitori, il sistema permette a runtime di agire opportunamente su questi parametri per mitigare eventuali attacchi di sicurezza.

I casi tipici di utilizzo della gestione del traffico sono:

- protezione API dai tipici attacchi DOS
- tuning del traffico API in funzione del carico gestibile dall'infrastruttura di back-end sottostante
- governo delle applicazioni fruitori di un API ed eventuali politiche di tracciamento e monetizzazione in funzione del traffico ammesso

Le politiche di gestione del traffico adottate sono:

- numero massimo di chiamate verso i sistemi di back-end (**maxTps**)
- livelli di sottoscrizione alle API da parte delle applicazioni (**policies**)

11.1 Throughput massimo del back-end

Il throughput massimo del back-end è la capacità di trasmissione che un servizio è in grado di reggere senza generare errori di carico.

Per evitare il sovraccarico del sistema di back-end, è possibile stabilire dei limiti di accesso, definiti con il parametro **Maximum Backend Throughput**, in fase di definizione di una API. L'unità di misura sono le TPS (transazioni per secondo) e indicano il numero di richieste che il sistema di back-end può gestire in un secondo. **È responsabilità del progetto effettuare i test di carico/performance per stabilire il carico che può sopportare il back-end.**

11.2 Livelli di sottoscrizione

I livelli di sottoscrizione determinano il limite di accessi disponibili per ogni applicazione che ha sottoscritto l'API. Insieme al throughput descritto sopra, definendo anche un livello di sottoscrizione adeguato, si può aumentare la protezione del back-end dall'eccessivo carico. In fase di pubblicazione delle API sul gateway si può scegliere di rendere l'API disponibile secondo uno dei livelli di sottoscrizione sottoelencati, ciascuno dei quali determina un preciso limite di accessi:

- il livello **Unlimited** che consente accesso illimitato all'API
- il livello **Gold** che, da impostazione default, consente a un'applicazione di accedere all'API fino a 5000 richieste al minuto
- il livello **Silver** che, da impostazione default, consente a un'applicazione di accedere all'API fino a 2000 richieste al minuto
- il livello **Bronze** che, da impostazione default, consente a un'applicazione di accedere all'API fino a 1000 richieste al minuto

- Il limite associato ai livelli gold, silver e bronze è l'attuale configurazione adottata, in fase di pubblicazione delle API è possibile concordare nuovi livelli di accesso in funzione delle esigenze di progetto

In caso di superamento dei limiti di richieste al minuto il fruitore riceve in risposta un errore con http status code *429 Too Many Requests*.