API: manuale utente

Sette best practice dei team API di successo

Manfred Bortenschlager

Director, Business Development for API-based Integration Solutions and API Management, Red Hat

Andrew Mackenzie

Director of Software Engineering API Management, Red Hat

Jaya Baskaran

Principal Technical Marketing Manager, Red Hat

Greg Pack

Senior Product Marketing Manager App Services and API Management Solutions, Red Hat

Contenuti

Riepilogo

I progressi nel moderno sviluppo applicativo, nella connettività e nelle infrastrutture, come le reti 5G e l'edge computing, modellano costantemente i modi e i luoghi in cui avvengono i processi aziendali. Tali progressi accelerano il business, ma non offrirebbero alcun vantaggio senza le interfacce di programmazione delle applicazioni (API).

Grazie alle API, che ormai rappresentano il tessuto connettivo digitale delle organizzazioni moderne, è possibile aggiungere nuove funzionalità a qualsiasi aspetto, dalle operazioni e dai prodotti alle strategie di collaborazione. Questo ebook è destinato a chi ha già predisposto una strategia API ma anche a chi si sta preparando a definirla e a presentarla e illustra a utenti e team le sette best practice alla base di un programma API di successo.

Introduzione

Prima di leggere l'ebook, è bene valutare gli obiettivi principali del programma API. In un secondo momento, alcune delle domande poste più sotto potranno contribuire a modificare, convalidare o fornire ulteriore sostegno a tali obiettivi.

Un programma API efficace deve prendere forma dalla strategia globale di un'organizzazione per poter contribuire al raggiungimento degli obiettivi aziendali. Una strategia ha le carte in regola quando risponde alle 4 domande seguenti:

1. Perché vogliamo implementare le API?
2. A quali utenti sono destinate queste API?
3. Quali risultati concreti vogliamo ottenere con queste API?
4. Come intendiamo eseguire il programma API per ottenere ciò che ci siamo prefissati?

Perché vogliamo implementare le API?

Per valutare le API è indispensabile comprenderne appieno il valore e la finalità. Per ottimizzare il valore che possono offrire all'azienda, è cruciale evitare i pregiudizi comuni. Uno tra i più diffusi presuppone che le API abbiano valore solo se vengono pagate dagli utenti, il che è vero solo se l'API è il prodotto. Molti progetti API restano però interni all'azienda e vengono usati per accrescere altri parametri, come la velocità di accesso ai mercati, la riutilizzabilità, le vendite, la consapevolezza del marchio o le segnalazioni degli affiliati.

Cinque tipici scenari di utilizzo dei fornitori di API includono:

1. **Scalabilità su ogni canale.** Estendi l'accessibilità tramite canali diversi, quali le app mobile, Internet of Things, ecc.
2. **Crescita dell'ecosistema.** Promuovi gli ecosistemi dei clienti o dei partner per incentivare l'espansione aziendale.
3. **Espansione della copertura.** Amplia le reti per le transazioni o la distribuzione dei contenuti.
4. **Adozione di nuovi modelli aziendali.** Promuovi l'innovazione tramite nuovi modelli di business.
5. **Innovazione interna.** Genera innovazione all'interno dell'organizzazione.

Alcune tra le più moderne società tecnologiche, come Amazon, Salesforce, Twilio, tra le altre, hanno adottato strategie API riuscite. Ciò non significa, tuttavia, che solo le grandi corporation possano utilizzare le API per garantirsi monetizzazione, innovazione, aumento delle collaborazioni o agilità. Molte aziende consolidate si avvalgono anche di sistemi tradizionali, che non possono essere sostituiti da un momento all'altro e che probabilmente non verranno mai abbandonati. Le API sono il collante perfetto per l'integrazione con i sistemi in uso, perché possono esporli all'interno o all'esterno e combinarli con altri servizi per creare nuovo valore. Nel proprio percorso verso la trasformazione digitale, qualsiasi azienda può trarre vantaggio dall'economia delle API se definisce una strategia API ben strutturata.

È chiaro che l'organizzazione deciderà di investire nelle API solo se i vantaggi saranno sufficientemente espliciti.

A quali utenti sono destinate queste API?

Per definire i parametri di successo del programma API è necessario capire chi saranno gli utenti finali delle API.

- **Utenti finali interni.** Per i team interni, le API sono strumenti per l'accesso alle informazioni aziendali necessarie per una serie di obiettivi di business.
- **Utenti finali esterni.** Le stesse API o i loro sottoinsiemi possono essere utilizzati nelle applicazioni realizzate da sviluppatori terzi.

È chiaro che l'organizzazione deciderà di investire nelle API solo se i vantaggi saranno sufficientemente espliciti.

Quali risultati concreti vogliamo ottenere con queste API?

Per identificare i risultati concreti ottenibili con le API, occorre considerare le prospettive interne ed esterne rispetto all'organizzazione.

- **Prospettiva interna.** Utilizza risorse esclusive e pertinenti alla tua azienda. Le organizzazioni che hanno informazioni distintive, come i dati sui "like" di Meta, possono offrire API preziose.
- **Prospettiva esterna.** Tieni conto delle dinamiche e delle tendenze del mercato, dei concorrenti e dei comportamenti dei clienti, perché sono forze esterne capaci di dar forma alle strategie aziendali e di influenzare le funzionalità delle API.

Le dinamiche di mercato incidono fortemente sul mapping delle API. Le prime aziende di mapping hanno sottovalutato la domanda in tempo reale, contribuendo all'ascesa di startup come Waze, poi acquistata da Google, che ne ha integrato la tecnologia in un'API di successo. Anche Twitter, Reddit e altri giganti hanno adottato le API per promuovere innovazione e crescita dell'ecosistema.

Spesso, la giusta strategia API mette insieme risorse interne e informazioni sul mercato esterno.

Come intendiamo eseguire il programma API per ottenere ciò che ci siamo prefissati?

I punti chiave di questa ultima domanda sono implementazione ed esecuzione, due aspetti che richiedono una pianificazione meticolosa.

1. **Scelte tecnologiche.** Determina lo stack tecnologico per la realizzazione delle API.
2. **Progettazione e manutenzione.** Prevedi le strategie di progettazione e manutenzione delle API.
3. **Promozione e marketing.** Promuovi l'impiego delle API all'interno dell'organizzazione e la loro commercializzazione all'esterno.
4. **Allocazione delle risorse.** Assegna le risorse necessarie allo sviluppo e alla manutenzione delle API.
5. **Composizione del team.** Organizza un team in grado di sviluppare e gestire le API.
6. **Community degli sviluppatori.** Crea e gestisci un piano di comunicazione dedicato agli sviluppatori interni ed esterni.
7. **Metriche del successo.** Definisci i metodi per tenere traccia del successo delle API rispetto agli obiettivi aziendali prefissati.

Facendo emergere le ragioni, le persone, i contenuti e le modalità delle API, sarà possibile promuovere innovazione e crescita aziendale, in un'economia delle API in continua evoluzione. Su questi aspetti, diversi per ogni organizzazione, incidono gli obiettivi, la strategia scelta e il team API designato, la cui importanza è cruciale.

Il team API

Un team che si occupa di API è in genere strutturato come qualsiasi altro team di prodotto, e deve occuparsi di creare, distribuire, gestire e ottimizzare l'infrastruttura da cui altre persone dipendono, a prescindere se i clienti sono interni o esterni.

Come i team di prodotto, i team API possono essere molto variegati, ma devono in genere includere una persona che agisca come responsabile della strategia e degli obiettivi e che concentri la sua attenzione sul prodotto, membri che si occupino della progettazione assicurandosi di adottare gli approcci migliori, ingegneri che codifichino la tecnologia API e personale operativo che gestisca l'esecuzione vera e propria delle API. Nel tempo è possibile aggiungere altre persone come membri del team di supporto e della community, specialisti delle API, responsabili della sicurezza e altri ancora. Il team allargato potrà anche includere membri della community di sviluppatori.

Soprattutto nelle organizzazioni più piccole, anche se il team prevede numerosi membri, alcuni possono rivestire più ruoli. È importante garantire che siano rappresentate le opinioni di tutte le parti interessate, anche se ciò significa che uno dei membri dovrà occuparsi di risolvere eventuali preoccupazioni degli altri.

I team API sono spesso temporanei, con partecipanti che appartengono a diverse unità organizzative e fanno capo a diversi responsabili. In queste condizioni, la definizione di una visione condivisa delle API può essere complicata. Ai programmi API più ampi possono partecipare diversi team API.

Qualunque sia la grandezza dell'organizzazione, le sette best practice descritte nell'ebook aiuteranno a organizzare un team API di successo, che potrà essere costituito da più persone rispetto a quanto previsto.

Best practice n. 1: attenzione illimitata al valore delle API

La priorità dei programmi API è erogare valore senza aumentare la complessità. Una delle misure del successo delle API è l'utilità per l'utente, determinata dalla proposta di valore. Per una commercializzazione efficace, quest'ultima deve essere convincente. Se la proposta è allineata agli obiettivi aziendali, garantisce la sostenibilità e ciò permette alle aziende consolidate di migliorare le proprie offerte tramite le API.

Il modello di valore delle API di Alex Osterwalder allinea i vantaggi erogati agli utenti alle caratteristiche dell'API, creando un ingranaggio perfetto tra le esigenze dell'utente e il valore dell'API.[1] È fondamentale che l'API soddisfi i bisogni, riduca le problematiche e generi valore.

Cosa significa in termini di API?

In questo processo iterativo, la prima fase descrive le attività che gli utenti devono completare, ad esempio, l'invio automatico di comunicazioni urgenti in caso di emergenza, il backup di file importanti, e l'analisi dei dati per individuare eventi specifici.

Successivamente, occorre identificare le criticità che gli utenti riscontrano prima, durante e dopo aver cercato di completare un processo: come il garantire più tentativi di invio dei messaggi, il rilevamento degli errori, la gestione di più messaggi, l'integrazione di sistemi di messaggistica basati sulla posizione, la protezione della consegna dei file con consumo minimo di larghezza di banda e la correlazione in tempo reale di grandi volumi di dati.

La terza fase nella creazione di un profilo utente consiste nel delineare i potenziali vantaggi, ad esempio l'invio di altri tipi di notifiche che creano opportunità invece di avvisi di minacce, eliminando apparecchiature di storage superflue se l'affidabilità è abbastanza buona, e l'avvio automatico di azioni sulla base degli eventi.

Passando alla mappa del valore, occorre mettere a punto la funzionalità, le caratteristiche e i servizi dell'API, concentrandosi su ciò che risolve le criticità e su ciò che produce valore. Questo processo genera esempi concreti, come un'API di messaggistica che garantisce la consegna dei messaggi, un'API per la sincronizzazione degli storage per aggiornamenti efficienti delle versioni e un'API di aggregazione dei dati che fornisca flussi di dati configurabili.

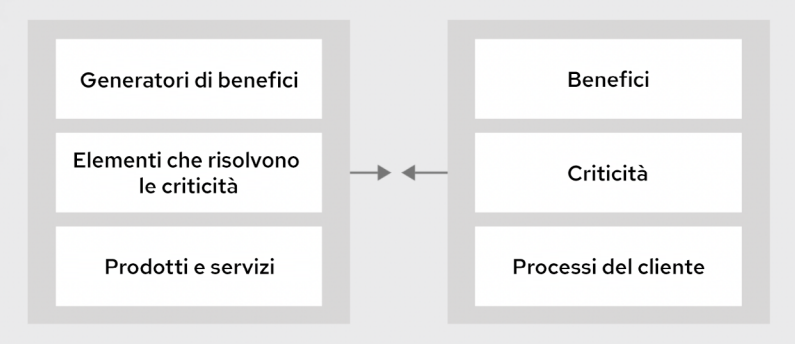

Figura 1. Struttura della proposta di valore

Infine, il team dovrà completare l'utile esercizio di comporre diverse dichiarazioni che chiariscono il legame tra l'API e il profilo utente. Il messaggio globale che riassume e astrae tutte le dichiarazioni di idoneità rappresenta la proposta di valore della tua API. Tali dichiarazioni irrobustiscono ulteriormente l'allineamento API-utente, consolidando la proposta di valore. Nel caso dell'esempio di API di messaggistica, la dichiarazione sarà simile a quanto segue:

> *"La citazione del cliente va collocata qui. Fare attenzione all'espansione del testo quando si aggiunge il contenuto della barra laterale. Alcune lingue, come il tedesco, occupano più spazio dell'inglese. Per poter inserire l'eventuale traduzione, 1/4 della barra laterale deve rimanere vuoto."*

Ciò potrebbe sembrare un lavoro eccessivo, perché l'API è prettamente interna. Nonostante la reazione sia naturale, occorre ricordare che l'attenzione al valore è fondamentale anche negli esempi di utilizzo interno. Se la proposta di valore non è ben definita, presentare l'API agli altri team risulta più complicato e dispendioso in termini di tempo. Una proposta di valore ben definita fa sì che il programma API contribuisca notevolmente al successo dell'azienda.

[1] Strategyzer, "Value proposition canvas." Accesso: settembre 2023.

Considerazioni sul valore del programma API

Il valore del programma API può essere definito con facilità valutando alcuni aspetti chiave:

1. **Identificazione dell'utente.** Identifica gli utenti in base alla relazione (clienti, partner e sviluppatori), al ruolo e alle preferenze.
2. **Individuare problematiche e vantaggi.** La struttura della proposta di valore nella Figura 1 è un riferimento per individuare le problematiche, i benefici e le principali esigenze dell'utente. Misura i miglioramenti delle metriche (velocità, fatturato e costi) e le potenziali nuove opportunità.
3. **Scenari di utilizzo supportati.** Utilizza la struttura delle proposta di valore per identificare gli elementi che risolvono le criticità e quelli che generano benefici. Progetta l'API affinché soddisfi questi specifici scenari di utilizzo.
4. **Valore futuro.** Pianifica la proposta di valore tenendo in conto il futuro, ovvero prevedendo variazioni imminenti, tendenze o innovazioni tecnologiche che possono sostenere la crescita del valore nel tempo.
5. **Valore interno all'organizzazione.** Valuta i vantaggi interni dell'API e il valore che può potenzialmente rappresentare per gli altri team.

Le risposte a queste domande possono costituire una chiara proposta di valore per il tuo programma API e garantire l'allineamento alle esigenze degli utenti e agli obiettivi dell'organizzazione.

Best practice n. 2: chiarire il modello aziendale fin dall'inizio

Per creare un'API di successo, la sola proposta di valore non basta. Occorre abbinare l'API a un modello aziendale ben definito. Benché riconoscere e trasmettere il valore delle API sia importante, esse generano anche costi che devono essere compensati da benefici monetari o comunque tangibili. Nel libro scritto a più mani "Business Model Generation", Alex Osterwalder definisce il modello aziendale di un'organizzazione come il modo in cui questa propone, crea, eroga e acquisisce valore.

I vantaggi di un modello aziendale di API

La struttura del modello aziendale esamina i componenti chiave del modello, che includono:[2]

1. **Proposta di valore.** Definisce il valore esclusivo dell'API.
2. **Flussi di fatturato.** Delinea il modo in cui l'API genera utili.
3. **Struttura dei costi.** I costi associati alla gestione dell'API.
4. **Segmento di clientela.** I gruppi di utenti target.
5. **Relazioni con i clienti.** Le modalità con cui l'azienda interagisce con gli utenti.
6. **Canali.** Metodi di distribuzione per raggiungere gli utenti.
7. **Partner principali.** Le collaborazioni strategiche per il successo dell'API.
8. **Attività principali.** Attività fondamentali necessarie per il funzionamento dell'API.
9. **Risorse principali.** Le risorse indispensabili per l'implementazione dell'API.

Figura 2. La struttura del modello aziendale

Utilizzando le risorse già esistenti, le API possono garantire nuove opportunità. Tuttavia, oltre a riconoscere il valore strategico delle API, è necessario anche gestirne i costi. Un modello non adatto all'azienda può far lievitare i costi dell'API, indipendentemente dal suo valore.

Considerazioni sul modello aziendale di API

Per allineare modello aziendale e utilizzo dell'API, vanno considerate 5 aree vitali:

1. **Valore dell'API per l'organizzazione.** Valuta diversi valori senza limitarti a quello monetario, approfondendo come l'API può contribuire a estendere la copertura o a promuovere l'innovazione in azienda.
2. **Acquisizione del valore.** Definisce il miglior modo per acquisire il valore identificato, riducendo gli ostacoli alla realizzazione del valore massimo.
3. **Copertura dei costi.** Individua i costi legati all'API, che in genere non si limitano al team API, ma includono quelli per i reparti di ingegneria o di marketing, e prevedi le necessarie coperture.
4. **Impegno a lungo termine.** Le API richiedono un impegno continuo per le attività di gestione e manutenzione, che va oltre l'investimento iniziale.
5. **Collaborazioni strategiche.** Identifica le partnership fondamentali e sfrutta le offerte complementari di partner e fornitori nella fase di sviluppo e di accesso al mercato dell'API.

Best practice n. 3: progettazione e implementazione incentrata sull'utente

Affinché possa fornire un'esperienza dell'utente coerente, la progettazione dell'API deve prevedere principi fondamentali. L'obiettivo è ottenere un livello di progettazione "pronto all'uso", ovvero API il cui utilizzo non deve essere spiegato a chi è esperto. La progettazione delle API deve perciò incentrarsi su due aree chiave: semplicità e flessibilità.

Semplicità

La semplicità della progettazione delle API dipende dal contesto, e può essere semplice per uno scenario di utilizzo e complessa per un altro. Ecco perché è necessario bilanciare quanto siano dettagliati i metodi API. La semplicità può essere valutata a vari livelli:

1. **Formato dei dati.** Scegli tra XML, JSON, formati proprietari o loro combinazioni.
2. **Struttura del metodo.** I metodi possono essere estremamente generici o molto specifici per rispondere a richieste mirate.
3. **Modello di dati.** Il modello di dati alla base può essere molto diverso rispetto a quello esposto tramite l'API, incidendo sull'usabilità e sulla manutenibilità.
4. **Autenticazione.** Meccanismi di autenticazione diversi hanno punti di forza e di debolezza diversi. Il più adatto dipende dal contesto.
5. **Policy di utilizzo.** I diritti e le quote degli sviluppatori devono essere facilmente comprensibili e gestibili.

Flessibilità

È fondamentale compensare semplicità e flessibilità. L'impiego di un'API estremamente semplice può essere limitato a condizioni di utilizzo molto specifiche, senza lasciare spazio ad altri esempi di utilizzo. Affinché un'API sia flessibile, occorre individuarne il potenziale spazio operativo, includendo i sistemi e modelli di dati alla base, e capire quali sottoinsiemi di operazioni sono praticabili e di valore. Per trovare il giusto equilibrio tra semplicità e flessibilità occorre:

1. **Cercare di esporre operazioni di dimensioni minime**, che combinate possono coprire l'intero spazio operativo.
2. **Identificare scenari di utilizzo comuni e di valore.** Progetta un secondo livello di meta-operazioni che combinano operazioni di dimensioni minime per servire questi esempi di utilizzo.

Considerazioni per la progettazione del protocollo API

Anche se lo stile architetturale REST (REpresentational State Transfer) resta lo standard di riferimento per lo sviluppo API, i protocolli API si vanno invece differenziando. Tra i più recenti citiamo Streaming API o WebSockets. Neanche le classiche API dei servizi web spariranno nell'immediato. Il principio su cui basare la scelta del protocollo è il seguente: individuare quello più rilevante per l'utente e quindi trovare il giusto equilibrio tra semplicità e flessibilità. L'infrastruttura per l'erogazione delle API REST è in continua evoluzione.

Durante la progettazione delle API, occorre considerare quanto segue:

1. **Allineamento agli scenari di utilizzo.** Dopo aver identificato gli scenari di utilizzo, progetta l'API in modo che sia in grado di supportarli, mantenendo al contempo la flessibilità per gli scenari più innovativi e meno frequenti.
2. **API RESTful, ma non a tutti i costi.** Le API RESTful sono senz'altro all'avanguardia, ma occorre valutare che siano davvero adatte alle esigenze organizzative, perché per alcune situazioni specifiche potrebbero essere idonei stili architetturali diversi.
3. **Astrazione del modello di dati.** L'API va implementata prevedendo un livello di astrazione tra l'API stessa e il modello di dati, così da evitare l'accesso diretto al database quando non è necessario.
4. **Considerazioni geografiche.** Valuta anche aspetti non funzionali come la latenza e la disponibilità, collocando i datacenter in modo strategico e in prossimità delle aree in cui risiede la maggior parte degli utenti.
5. **Integrazione con i prodotti.** La progettazione delle API deve essere sincronizzata con gli altri prodotti tramite attività di coordinamento o disaccoppiamento, e comunicando sempre chiaramente ogni decisione alle parti interessate, interne ed esterne.

Infrastruttura di implementazione e deployment delle API

Con il passaggio al cloud e alle infrastrutture di cloud ibrido per i dati e le applicazioni, anche le infrastrutture di implementazione e deployment delle API si diversificano. Esistono API realizzate a partire da microservizi e combinate con altre API realizzate come monoliti, e tutte le varietà possibili tra questi due estremi. Possono trovarsi nei container, su macchine virtuali, su bare metal o in un ambiente di cloud pubblico.

Come accade con qualsiasi altra applicazione, per una gestione ottimale del ciclo di vita delle API è fondamentale disporre di una pipeline automatizzata di integrazione e distribuzione continue (CI/CD). L'adozione di GitOps e di Argo CD può garantire la centralità di gestione della configurazione delle API, la distribuzione continua automatizzata e semplificare la collaborazione tra i team API, il che si traduce in uno sviluppo più agile e in API di migliore qualità.

Best practice n. 4: mettere al primo posto l'operatività delle API

Una volta che le API sono attive, il team responsabile della piattaforma API si accerta che siano accessibili ed esposte secondo le aspettative degli sviluppatori. Esistono diverse soluzioni adatte a questo scopo, ma innanzitutto è fondamentale selezionare la strategia di gestione delle API più adatta, che dovrà avere due obiettivi principali:

1. Ottimizzare i processi interni per renderli efficienti e ridurre i costi.
2. Garantire l'efficienza delle operazioni per soddisfare le aspettative degli sviluppatori esterni che partecipano al programma.

L'articolo Building Great APIs Part 1: The Gold Standard e il grafico a ciambella della Figura 3 illustrano come raggiungere questi obiettivi.

Grafico a ciambella dell'operatività delle API

Il grafico a ciambella aiuta a definire le tattiche operative per realizzare la strategia API dell'organizzazione. Il cerchio interno della ciambella rappresenta le attività e gli effetti interni all'organizzazione, mentre tutto ciò che è esterno all'anello mostra gli effetti esterni.

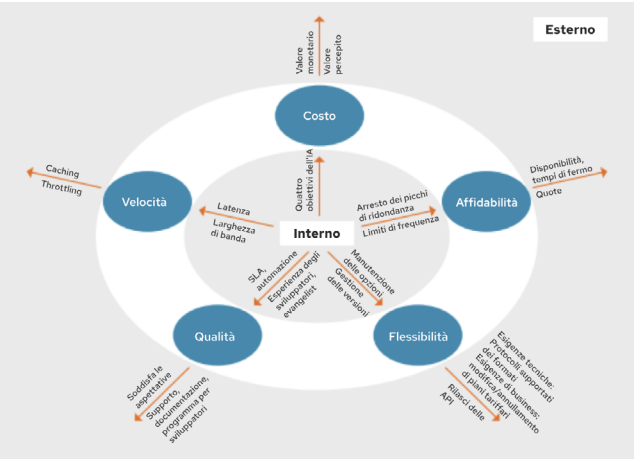

Figura 3. Grafico a ciambella dell'operatività delle API

1. **Affidabilità.** La disponibilità delle API può essere garantita applicando ridondanza, quote o limiti di frequenza allineati ai modelli aziendali, in modo da evitare i tempi di fermo.
2. **Flessibilità.** È bene offrire opzioni tecniche e di adozione, ad esempio possibilità di modificare piani tariffari e cancellazioni, benché questa flessibilità possa incrementare l'impegno e i costi interni.
3. **Qualità.** Garantisci il rispetto coerente delle aspettative degli sviluppatori con accordi sul livello di servizio (SLA) e processi semplificati.
4. **Velocità.** Per ottenere bassa latenza ed elevata larghezza di banda è possibile avvalersi di tecniche come il throttling e il caching, meglio se allineate ai modelli aziendali.
5. **Costo.** È indispensabile ottimizzare il valore degli sviluppatori riducendo al minimo i costi interni, ma senza compromettere la qualità.

Numerosi fornitori, inclusa Red Hat, offrono infrastrutture tecnologiche capaci di risolvere molte di queste sfide operative. Affidarsi a un fornitore è spesso conveniente, ma occorre disporre di una strategia meticolosa.

Componenti essenziali per la gestione delle API

Per far funzionare un ecosistema di API servono componenti unici che ne garantiscano l'efficienza di gestione. Sebbene possono variare a seconda della strategia API scelta, i tre componenti principali saranno sostanzialmente gli stessi:

1. **Controllo dell'accesso.** Adotta sistemi di autenticazione e autorizzazione per permettere l'accesso e l'identificazione del traffico in entrata.
2. **Limiti di frequenza e criteri di utilizzo.** Applica quote e limitazioni al traffico per prevederne il carico.
3. **Analisi.** Acquisisci e analizza i modelli di traffico per monitorare l'utilizzo delle API.

Considerazioni per l'operatività delle API

Una strategia per l'operatività delle API armonizzata con le altre strategie generali dell'organizzazione facilita l'assegnazione delle risorse in funzione dell'importanza dell'API.

Durante la progettazione del piano per l'operatività delle API, occorre considerare quanto segue:

1. **Controllo dell'accesso.** Definisci gli utenti che possono accedere ed eseguire azioni, applicando dei limiti per garantire un utilizzo delle API attento alla sicurezza.
2. **Metriche e avvisi.** Puoi ottenere più visibilità adottando strumenti di analisi, misurazione di metriche personalizzate e impostazioni di avvisi relativi alle prestazioni delle API.
3. **Gestione dei picchi.** Prevedi un'infrastruttura con controllo degli accessi e criteri e imposta meccanismi di fallback come l'interruzione o il throttling dei picchi.
4. **Responsabilità dell'uptime delle API.** La trasparenza della titolarità dell'uptime delle API è fondamentale, poiché è direttamente connessa alla generazione e all'acquisizione di valore.
5. **Mitigare l'utilizzo improprio.** Occorre distinguere tra l'utilizzo improprio prevedibile o imprevedibile, e gestire queste evenienze con operazioni proattive, termini e condizioni.
6. **Comunicazione.** È indispensabile predisporre un piano di comunicazione trasparente rivolto agli sviluppatori interni ed esterni, per informarli di interruzioni pianificate, interventi di manutenzione, modifiche alle API.

Best practice n. 5: ossessionati dall'esperienza degli sviluppatori

L'esperienza degli sviluppatori, benché possa far pensare alla progettazione delle API, è ben diversa. Possiamo immaginarla come il confezionamento e la distribuzione dell'API, invece che l'API stessa. Potremmo anche avere un'API REST o containerizzata progettata in modo straordinario, ma se accedervi, leggerne la documentazione o testarla sono attività complicate, l'esperienza dello sviluppatore sarà mediocre e inciderà in modo negativo sul successo dell'API.

Per il successo dell'API una buona progettazione non basta

Se gli sviluppatori hanno difficoltà a interagirvi e quindi ad adottarla, anche un'API la cui progettazione offre semplicità e flessibilità è sprecata. Una progettazione ben ponderata, invece, incide nettamente sull'esperienza dello sviluppatore e sull'adozione dell'API, che è una parte fondamentale dell'esperienza complessiva.

John Musser, uno dei pionieri delle API e della loro gestione, ha descritto diversi modalità per promuovere l'interazione degli sviluppatori, che sono valide ancora oggi:

1. Definire chiaramente la finalità delle API.
2. Garantire l'accesso gratuito e la registrazione semplificata.
3. Comunicare le tariffe in modo trasparente.
4. Fornire una documentazione approfondita.

Una metrica che permette di migliorare la progettazione di API semplificandone l'adozione è il Time To First Hello World (TTFHW), che misura l'interazione iniziale con l'API. La metrica Time To First Profitable App (TTFPA) coglie invece un contesto più ampio, ponendo più enfasi sui programmi per gli sviluppatori. Questa metrica è più complicata perché la parola "profitable" (redditizia) cambia definizione a seconda dell'API e della strategia aziendale. Prendere in considerazione questo aspetto è utile perché forza a pensare all'operatività dell'API nel quadro più ampio del programma API.

I due principi alla base dell'esperienza degli sviluppatori sono la progettazione di un prodotto o servizio di evidente valore e la facile accessibilità. Un programma di coinvolgimento degli sviluppatori, che preveda portale, community, evangelist, eventi e misurazioni, promuoverà la loro interazione con le API. Alcuni esempi di metriche tipiche di un portale degli sviluppatori sono il numero di visite alle pagine, le registrazioni, il traffico API o le richieste di supporto. Gli eventi possono essere misurati in base al numero di partecipanti, di adozioni delle API durante gli hackathon o di lead ottenuti. È utile anche la creazione di correlazioni, ad esempio il numero di registrazioni generate nel corso di una conferenza in un evento.

Un programma per sviluppatori dovrebbe includere i seguenti elementi:

- **Creazione di una community.** Eventi come gli hackathon promuovono le interazioni e le adozioni.
- **Evangelist sviluppatori.** Fondamentali per il successo dell'API, ne promuovono i vantaggi stimolando le interazioni.
- **Partner pilota e casi cliente.** I primi ad aver adottato le API possono fornire feedback e casi cliente di esempio.
- **Partner dell'ecosistema.** Le collaborazioni amplificano in modo sinergico l'adozione delle API.
- **Misurazione.** Metriche allineate agli obiettivi delle API contribuiscono all'efficienza delle attività di gestione.
- **Comunicazione.** Può avvenire tramite portali, newsletter, server Slack o Discord privati.

Costruire un programma per sviluppatori personalizzato in base al pubblico di riferimento promuove il coinvolgimento e migliora l'esperienza dei partecipanti.

Considerazioni per la valutazione dell'esperienza degli sviluppatori

Fornire agli sviluppatori un'esperienza ben ponderata può incentivarli a offrire il massimo delle proprie potenzialità. Le sei considerazioni seguenti sono fondamentali per valutare tale esperienza e creare API che non siano solo righe di codice, ma uno stimolo alla creatività e un incentivo alla produttività.

1. **Spiegazione del valore.** Sviluppa un breve "elevator pitch" che comunichi il valore delle API agli sviluppatori.
2. **TTFHW e TTFPA.** Valuta e riduci al minimo il tempo di realizzazione della prima applicazione redditizia, tenendo conto di tutti gli elementi che fanno parte dell'esperienza dello sviluppatore (come i portali).
3. **Processo di registrazione.** Le procedure di onboarding devono essere allineate agli scenari di utilizzo delle API, semplici e dirette per accelerare il successo dello sviluppatore.
4. **Proposta di valore.** Accertati che il valore dell'API sia chiaro, per attirare e mantenere gli sviluppatori.
5. **Supporto agli sviluppatori.** Dai priorità al supporto self service fornendo documentazione, risposte alle domande frequenti, forum e meccanismi aggiuntivi per risolvere i problemi più complessi.
6. **Utilizzi diversi.** Offri supporto e documentazione mirati agli sviluppatori che studiano scenari di utilizzo non convenzionali delle tue API.

Best practice n. 6: guardare oltre alla commercializzazione delle API

Il marketing delle API presso gli sviluppatori non è un'attività semplice, soprattutto se il valore presentato non corrisponde alle loro esigenze tecniche o di business. Il modello di commercializzazione STP (segmentazione, targeting e posizionamento) è valido per le API come per qualsiasi altro prodotto. Un marketing efficiente abbina l'API corretta agli sviluppatori più idonei, in base al valore dell'API e ai principi STP.

1. **Segmentazione.** Suddividi i clienti in categorie quali utenti interni, partner, utenti finali o sviluppatori esterni. Per un coinvolgimento efficace, l'ampio mercato degli sviluppatori va segmentato in base all'attività che deve essere svolta.
2. **Targeting.** Valuta l'attrattività del segmento in base all'accessibilità, alla sostanzialità e alla differenziazione. Le tattiche di marketing potranno quindi essere personalizzate per i segmenti più promettenti.
3. **Posizionamento.** Fai in modo che le tue API emergano risolvendo esigenze specifiche e criticità importanti e fornendo benefici ai segmenti di sviluppatori selezionati.

Dare priorità all'esperienza degli sviluppatori nelle iniziative di marketing è fondamentale per il successo. Strategie che prevedono l'impiego di evangelist, portali di sviluppo efficienti, hackathon e altri eventi possono creare e consolidare le relazioni con gli sviluppatori.

Considerazioni per la commercializzazione delle API

1. **Pubblico di riferimento.** Dai priorità ai gruppi di utenti principali, facendo in modo che le API si adattino a diverse fasi e possano essere utilizzate da utenti interni, partner, clienti o il pubblico in generale.
2. **Specialisti scelti.** Individua esperti allineati alla proposta di valore delle tue API, che siano in grado di promuoverle in modo efficace nei reparti di ingegneria, assistenza, vendite e gestione prodotto.
3. **Strategia degli eventi.** Seleziona le tipologie di evento (segmento orizzontale o verticale, a livello globale o locale, conferenza o hackathon) in base agli obiettivi delle tue API.
4. **Idoneità degli hackathon.** Valuta la pertinenza degli hackathon (registrazioni, download di SDK, applicazioni, attività di selezione, branding) e programmali di conseguenza.
5. **Marketing interno.** Assicurati il supporto nelle diverse unità, chiarisci ogni aspetto con il reparto marketing e comunica i vantaggi ai team di prodotto e ai clienti.

Best practice n. 7: non dimenticare la dismissione delle API e la gestione delle modifiche

Le linee guida per le API danno particolare importanza alle fasi di progettazione, creazione e operatività. In realtà, una delle fasi più critiche e spesso trascurate del ciclo di vita delle API si ha mesi dopo il rilascio e l'inizio delle operazioni, ovvero la gestione degli aggiornamenti delle API e il ritiro quando l'API raggiunge l'obsolescenza.

Le interruzioni causate da modifiche improvvise possono intaccare la fiducia degli utenti e generare costi significativi, soprattutto se si deve ricorrere a sviluppatori sconosciuti, si devono approvare le applicazioni mobili o sono presenti dispositivi mobili a cui mancano le capacità di aggiornamento.

Le modifiche alle API possono causare o meno interruzioni. Non causano interruzioni l'aggiunta di migliorie o nuovi metodi. Al contrario, rimozioni, modifiche o il ritiro completo provocano interruzioni operative. Le versioni principali e i piani di migrazione sono progettati per gestire le modifiche più importanti, mentre le versioni secondarie gestiscono quelle che non causano interruzioni. Accertarsi che le modifiche non abbiano impatto sul funzionamento delle applicazioni può essere complesso; per questo qualsiasi cambiamento alle API, anche se di minore entità, dovrebbe essere eseguito prevedendo:

- Un endpoint di test, nel quale installare la nuova versione prima del lancio.
- L'invio di un'email o comunicazione di altro tipo agli sviluppatori, che dia informazioni sulla modifica e sulle relative tempistiche.

Comunicazioni efficaci e contratti trasparenti sono di importanza vitale. È bene condividere informazioni sui problemi, rispettare gli impegni e definire la durata del supporto offerto alla versione. Ad esempio, il ritiro di un'API richiede un approccio strutturato che dovrà prevede informative dettagliate, copertura mediatica, piani di migrazione e strumenti per l'esportazione dei dati, se richiesti.

Un piano di migrazione facilita gli aggiornamenti delle API se:

1. Presenta la nuova versione da testare.
2. Notifica agli utenti l'imminente ritiro della versione precedente.
3. Offre assistenza durante la transizione.
4. Semplifica il ritiro della versione obsoleta.

Una gestione a 360° delle API include l'anticipazione di aggiornamenti e dismissioni, una comunicazione efficace e il consolidamento della fiducia degli utenti agendo in modo trasparente.

Considerazioni per la longevità delle API

1. **Garantire il proprio impegno al rispetto della garanzia sottoscritta dal cliente.** È innegabilmente l'aspetto più critico del programma API: quale livello di stabilità del servizio ti impegni a offrire ai tuoi utenti? È necessario definirlo chiaramente, perché questo impegno influenza l'adozione delle API.
2. **Elaborazione delle modifiche che causano o meno interruzioni.** Definisci le procedure di rilascio che consentono di rispettare la garanzia sottoscritta. Identifica le parti coinvolte e le fasi di approvazione.
3. **Comunicazione delle modifiche.** Individua, documenta e comunica le variazioni apportate utilizzando formati di definizione delle API. Garantisci la compatibilità e comunicazioni chiare.
4. **Gestione delle versioni.** Monitora l'utilizzo delle versioni meno recenti utilizzando gli ID degli sviluppatori e degli utenti e stabilisci una procedura di dismissione che eviti i problemi.
5. **Allineamento del prodotto.** Coordina in parallelo l'evoluzione delle API e le modifiche al prodotto, tenendo conto degli impegni del cliente e dei necessari adeguamenti.

Sostenere la strategia API aziendale

Le best practice qui delineate sono pensate per aiutarti a definire, mettere in atto e promuovere la tua strategia API. Il loro obiettivo principale è di adeguare la strategia alle nuove opportunità individuate per le API. Molte delle problematiche delineate nelle sezioni precedenti richiederanno ulteriori approfondimenti, e nel tempo emergeranno nuove opportunità o rischi. Ad esempio:

- Ci sono modalità per ampliare il valore dell'API?
- Le prospettive future sono state esaminate in modo sufficiente?
- La proposta di valore è sufficientemente attraente per coinvolgere davvero gli sviluppatori?

Di pari passo all'evoluzione del panorama IT, anche le API e la tua offerta devono guardare al futuro. I quattro principali stimoli al cambiamento sono:

1. **Forze di settore.** Nuovi concorrenti o servizi che possono sostituire le tue API.
2. **Forze di mercato.** Variabilità della domanda o delle condizioni del segmento di mercato.
3. **Forze macroeconomiche.** Variazioni del mercato globale che incidono sul budget a disposizione degli utenti.
4. **Trend.** Requisiti normativi o tecnologie in evoluzione.

Per un programma API duraturo

Il successo dell'avvio, della crescita e dell'evoluzione di un programma API si fondano sulla capacità di abbinare scenari di utilizzo consolidati ai potenziali clienti. Un programma API funzionale supporta l'innovazione grazie ad API flessibili che affrontano in modo proattivo scenari di utilizzo complessi attraverso la progettazione e l'operatività. È inoltre fondamentale che i termini e le condizioni delle tue API contengano gli strumenti necessari per agire in caso di comportamenti imprevisti.

Potresti dover riconsiderare la strategia API se:

- La dipendenza da un'innovazione imprevista offusca il valore della strategia. Per il successo dell'approccio è fondamentale una solida infrastruttura.
- La mancanza di scenari di utilizzo interessanti per te e per gli utenti indica che è necessario un diverso approccio.
- All'interno, sussistono dubbi sui possibili comportamenti negativi, che segnalano misure o comunicazione insufficienti.
- Frequenti compromissioni o utilizzi impropri delle API rivelano il mancato allineamento tra il valore previsto e quello effettivo.

Le API non vivono in un contesto isolato

La continua evoluzione del panorama tecnologico ha prodotto una serie di strumenti per lo sviluppo di applicazioni cloud native, compresi Kubernetes, Red Hat® OpenShift® e Red Hat OpenShift Service Mesh, che lavorano in sinergia per migliorare la connettività e accelerare lo sviluppo di applicazioni di altissima qualità. Tutti questi progressi, tuttavia, non mettono in discussione i principi alla base di una strategia API solida e l'importanza di un efficiente sistema di gestione delle API. Chi intraprende questo percorso deve cercare una soluzione di gestione delle API che sia allineata alla strategia aziendale complessiva e, al contempo, dimostri una profonda conoscenza o un'integrazione coerente con queste piattaforme, per permettere che la visione strategica aziendale possa diventare realtà.

Perché scegliere Red Hat per la creazione e la gestione delle API?

A sostegno del successo del tuo programma API, Red Hat propone un approccio alle API che mette al primo posto la progettazione e abbraccia l'intero ciclo di vita delle API, dalla pianificazione iniziale (includendo l'assegnazione delle risorse e la modellazione dello scenario di business), alla gestione delle API (con controllo dell'accesso, accesso alle API e analisi del loro utilizzo).

Per sostenere i team di sviluppo nell'adozione delle best practice specifiche dell'approccio che abbiamo analizzato in questa guida, Red Hat rende disponibile Red Hat 3scale API Management, la soluzione per la gestione delle API progettata appositamente per gli ambienti di cloud ibrido. Questa soluzione distribuita, containerizzata, leggera e cloud native consente di gestire carichi di lavoro di grandi dimensioni e promuove un approccio incentrato sulla sicurezza e sulla collaborazione. La condivisione e il controllo dell'accesso a servizi, risorse, applicazioni e sistemi aziendali in ambienti cloud pubblici e privati favoriscono la tua attività anche negli ecosistemi più complessi.

Tuttavia, poiché come abbiamo già visto le API non vivono in un contesto isolato, per adottare anche solo alcune delle best practice delineate è indispensabile una piattaforma completa per lo sviluppo di applicazioni cloud native. Red Hat 3scale API Management, inclusa nell'offerta Red Hat Application Foundations, è una raccolta di strumenti middleware che include funzionalità di integrazione, progettazione delle API, governance e registro delle API, streaming e messaggistica progettate per funzionare su Red Hat OpenShift Container Platform. Insieme, Red Hat OpenShift e Red Hat Application Foundations offrono alle organizzazioni una piattaforma applicativa cloud native completa per la distribuzione efficiente e sicura di nuovo software agli utenti, che garantisce loro anche opportunità, vantaggi e funzionalità strategiche.

Scopri nuovi livelli dello sviluppo di applicazioni cloud native con Red Hat 3scale API Management, Red Hat Application Foundations e Red Hat OpenShift e sperimenta l'approccio del futuro alla creazione semplificata, efficiente e collaborativa delle applicazioni.

Conclusioni

Speriamo che queste best practice contribuiscano a semplificare il tuo processo decisionale e diano risposte alle tue domande. Il percorso verso la costruzione della strategia API inizia con una comprensione completa, coerente e a livello dell'intera organizzazione del valore delle tue API. Tuttavia, il valore e gli obiettivi di business delle tue API non sono fissi. Il mondo cambia, e la strategia API deve adeguarsi di conseguenza.

L'esperienza maturata lavorando con i clienti e osservando l'economia delle API ci permette di cogliere alcuni aspetti chiave che caratterizzano un programma API di valore:

- Gli utenti adottano un'API per il valore che può offrire loro, perché è in grado di risolvere una criticità o di generare un beneficio.
- Un'API continua a fornire valore ai propri utenti quando offre una proposta di valore e una strategia che variano insieme al contesto.
- Il valore di un'API è riconosciuto all'interno dell'organizzazione, perché facilita l'esecuzione di operazioni importanti e aiuta a raggiungere obiettivi significativi.
- Un numero molto elevato di parti interessate (possibilmente tutte) si ritiene soddisfatta.

Scopri di più su Red Hat 3scale for API management ed esplora le risorse per la gestione delle API di Red Hat.