Platform Engineering - Maturity Model

Presentato da Cloud Native Computing Foundation Working Group Platforms

Versione 1.0.0
Rilasciato 31 Ottobre 2023
Traduzione: William Rizzo
Revisione: Nicolò Cambiaso Erizzo

Introduzione

Il white paper iniziale della CNCF sulla “Definizione delle Piattaforme” descrive cosa sono le Piattaforme Interne (Internal Platforms) per lo sviluppo cloud native e il valore che promettono di portare alle imprese. Per raggiungere quel valore, tuttavia, un’organizzazione deve riflettere su come perseguire deliberatamente i risultati e le pratiche di maggiore impatto e rilievo per il proprio contesto, tenendo presente che si vorrà affidare a una piattaforma interna che risponda appositamente ai propri bisogni - anche se quella piattaforma sarà magari solo documentazione su come utilizzare servizi di terze parti. Questo modello di maturità (Maturity Model) fornisce un quadro per favorire questa riflessione e per aiutare a identificare le opportunità di miglioramento in qualsiasi organizzazione.

Cosa è il Platform Engineering?

Ispirate alla cooperazione funzionale incrociata promessa dal DevOps, le piattaforme — e con esse la disciplina del Platform Engineering — sono emerse nelle imprese come una forma esplicita di articolare tale cooperazione. Le piattaforme curano e offrono la fruizione di funzionalità comuni, framework ed esperienze. Nel contesto di questo gruppo di lavoro e delle pubblicazioni correlate, l'attenzione è rivolta in particolare alle piattaforme che facilitano e accelerano il lavoro di utenti interni come i team di prodotto e di sviluppo applicativo.

Il Platform Engineering è la pratica di progettare e fornire tali piattaforme informatiche a sviluppatori e utenti interni, e comprende tutti gli aspetti delle piattaforme e le loro capacità — le loro persone, i processi, le policy e le tecnologie; così come i risultati aziendali desiderati che le guidano.

Per avere un contesto completo, si raccomanda di leggere il white paper della CNCF sulla Definizione delle Piattaforme (https://tag-app-delivery.cncf.io/whitepapers/platforms/).

Come utilizzare questo modello

Con l'aumento dell'importanza del Platform Engineering negli ultimi anni, sono emersi alcuni schemi ricorrenti. Organizzando questi schemi e le nostre osservazioni in un modello progressivo di maturità, miriamo ad orientare i team di piattaforma verso le sfide che potrebbero affrontare e le opportunità da perseguire. Ogni aspetto è descritto da un continuum di caratteristiche dei diversi team e delle organizzazioni ad ogni livello all'interno dell'aspetto. Ci aspettiamo che i lettori si ritrovino nel modello e identifichino opportunità nei livelli adiacenti.

Da notare che ogni livello aggiuntivo di maturità è accompagnato da maggiori requisiti di finanziamento e tempo delle persone. Pertanto, raggiungere il livello più alto non dovrebbe essere un obiettivo di per sé. Ogni livello descrive qualità che dovrebbero apparire in quella fase. I lettori devono considerare se la loro organizzazione e il loro contesto attuale beneficerebbero di queste qualità, dato l'investimento richiesto.

Tenete presente che ogni aspetto del Maturity Model è destinato ad essere valutato ed evoluto indipendentemente. Tuttavia, come in qualsiasi sistema socio-tecnico questi aspetti sono complessi e interrelati. Così, potreste scoprire che per migliorare in un aspetto dovete raggiungere un livello minimo anche in un altro aspetto.

È inoltre importante riconoscere che le implementazioni delle piattaforme variano da organizzazione a organizzazione. Assicuratevi di valutare lo stato attuale della trasformazione Cloud Native del *vostro* gruppo. Una risorsa fenomenale da sfruttare per questa valutazione è il Cloud Native Maturity Model (Modello di Maturità Cloud Native).

Infine, questo modello incoraggia le organizzazioni a maturare la loro disciplina di Platform Engineering e le piattaforme risultanti attraverso una pianificazione intenzionale. Tale pianificazione e disciplina sono di per sé un requisito per lo sviluppo di piattaforme mature e per l'evoluzione continua.

In generale, tenete presente che mappare la vostra organizzazione in un modello ne cattura lo stato attuale per *abilitare* l'iterazione progressiva e il miglioramento. [Martin Fowler] scrive: "*Il vero risultato di una valutazione del modello di maturità non è il livello in cui vi trovate ma l'elenco delle cose su cui lavorare per migliorare. Il vostro livello attuale è semplicemente un pezzo di lavoro intermedio per determinare quell'elenco di competenze da acquisire successivamente.*" In tale ottica, cercate di ritrovarvi nel modello, per poi identificare opportunità nei livelli adiacenti.

Contesto dietro a questo lavoro

È importante comprendere il contesto in cui un documento è stato scritto. Le seguenti sezioni delineano sommariamente il contesto dietro al modello, così come alcune aspettative per i lettori.

Pubblico di riferimento

Ogni lettore porta con sé un contesto unico e trarrà insegnamenti unici da questo modello. Di seguito sono elencate alcune *persone* che abbiamo in mente, assieme alle loro possibili motivazioni per interagire con questo modello:

- **CTO, VP e direttori della tecnologia**: Leader che cercano di tracciare una strada verso la trasformazione digitale e una maggiore produttività degli sviluppatori.
- **Responsabili dell'ingegneria**: Gruppi e individui che cercano di potenziare gli ingegneri per fornire valore con meno oneri e maggiore efficienza.
- **Enterprise architects**: Individui che si orientano nel paesaggio tecnologico moderno e disegnano una prospettiva orientata al valore e alla soluzione dei problemi tecnologici.
- **Platform Engineers e Product Manager di piattaforma**: Team e persone che cercano di costruire la migliore esperienza possibile per i costruttori di piattaforme e gli utenti di piattaforme.
- **Fornitori e responsabili di progetto**: Organizzazioni e ingegneri che desiderano progettare strumenti per consentire agli utenti di avere successo con piattaforme e funzionalità.
- **Sviluppatori di applicazioni e prodotti**: Utenti di piattaforma che cercano di comprendere più dettagliatamente cosa possono aspettarsi da una piattaforma interna.

Comprensione dei livelli

Questo modello non è inteso per classificare un'organizzazione o un team di piattaforma come interamente "Livello 1" o "Livello 4". Ogni aspetto dovrebbe essere considerato indipendentemente dagli altri; le caratteristiche di ciascun livello rappresentano un continuum all'interno di quell'aspetto ma non sono necessariamente accoppiate ad altri aspetti allo stesso livello. Ancora di più, molte organizzazioni vedranno caratteristiche di più di un livello applicabili ai loro team e al loro lavoro. Questo perché nessun livello è intrinsecamente buono o cattivo, solo contestuale agli obiettivi del team.

Le etichette per ciascun livello hanno lo scopo di riflettere l'impatto del Platform Engineering nella vostra organizzazione. Riconoscendo la vostra organizzazione a un dato livello, guadagnerete intuizioni sulle opportunità che seguono ai livelli successivi. I livelli con numerazione più bassa comprendono soluzioni più tattiche, mentre quelli con numerazione più alta sono più strategici.

Questo produce un potenziale processo per lo sviluppo e la maturazione della piattaforma che è simile allo sviluppo di altri prodotti digitali: occorre prima riconoscere un problema e la necessità di una nuova soluzione, poi sviluppare prodotti minimamente validi (MVP) come soluzioni ipotizzate, terzo iterare per risolvere al meglio il problema e assicurarsi che si adatti ai vostri clienti e infine scalare e ottimizzare il prodotto per risolvere il problema per molti team e utenti.

Simile al Modello di Maturità Cloud Native CNCF, questo modello evidenzia che i risultati aziendali di successo possono essere raggiunti solo bilanciando persone, processi e policy insieme alla tecnologia. In particolare, questo modello introduce aspetti che spesso non sono completamente di competenza di un singolo team interno, ma richiedono piuttosto la cooperazione attraverso il dipartimento di ingegneria e molto spesso l'organizzazione più ampia.

Ma non sembra adattarsi

Questo è perfettamente normale! Tutte le organizzazioni e gruppi hanno dinamiche e parametri specifici.

Tenete presente che l'obiettivo di questo documento non è prescrivere una formula rigida, ma piuttosto un quadro che potete applicare alle vostre circostanze. Non ogni singola parola potrebbe essere rilevante per voi, ma speriamo che il contenuto vi ispiri a riflettere sul vostro, specifico, viaggio nella piattaforma, prendendo ciò che ha senso e lasciando il resto.

L'obiettivo di questo modello è fornire uno strumento per aiutare e guidare i praticanti dell'ingegneria delle piattaforme, gli stakeholder e altre parti interessate. La progettazione e l'implementazione delle piattaforme non è una scienza esatta, ma dipende piuttosto dalle esigenze di progetto di ogniorganizzazione in quel particolare momento e luogo.

Tabella del Modello

Aspetto		Provvissiorio	Operativo	Scalabile	Ottimizzazione
Investimento	*Come vengono allocati il personale e i fondi alle capacità della piattaforma?*	Volontario o Temporaneo	Team Dedicato	Come Prodotto	Ecosistema Abilitato
Adozione	*Perché e come gli utenti scoprono e utilizzano le piattaforme interne e le funzionalità della piattaforma?*	Erratica	Spinta Estrinseca	Trazione Intrinseca	Partecipativa
Interfacce	*Come gli utenti interagiscono e consumano le funzionalità della piattaforma?*	Processi Personalizzati	Strumentazione Standard	Soluzioni Self-Service	Servizi Integrati
Operatività	*Come vengono pianificate, priorizzate, sviluppate e mantenute le piattaforme e le loro funzionalità?*	Su Richiesta	Tracciato Centralmente	Abilitato Centralmente	Servizi Gestiti
Misurazione	*Qual è il processo per raccogliere e incorporare feedback e apprendimenti?*	Ad Hoc	Raccolta coerente	Insights	Quantitativa e Qualitativa

Modello in Dettaglio

Investimento

Come vengono allocati il personale e i fondi alle funzionalità della piattaforma?

L'investimento nelle piattaforme e nell'ingegneria delle piattaforme è il processo di allocazione del budget e delle persone per costruire e mantenere funzionalità comuni. È comune che le iniziative vengano descritte come costruite organicamente dal basso verso l'alto, o guidate attraverso iniziative dall'alto verso il basso. In entrambi i casi, è la capacità di investire uno sforzo sostenuto che guida il lavoro ad alto impatto. Questo aspetto cattura come la scala e l'ampiezza dell'investimento possono impattare sul successo della piattaforma.

Livello 1, Provvisorio — Volontario o temporaneo

Possono esistere capacità individuali per fornire fondamenta comuni per funzionalità comuni o critiche. Queste capacità sono costruite e mantenute per necessità piuttosto che pianificate e finanziate intenzionalmente.

Queste capacità sono costruite e mantenute da persone assegnate temporaneamente o volontariamente; non ci sono finanziamenti o personale centrali assegnati intenzionalmente a loro. Dipendono dai requisiti tattici attuali dei loro utenti.

Caratteristiche:

- Team "Hit" o "Tiger" sono creati per affrontare requisiti urgenti. Queste squadre hanno vita breve e non sono assegnate permanentemente né ricevono il tempo per fornire pianificazione e supporto a lungo termine.
- Migrazioni, miglioramenti o potenziamenti sono spesso considerati lavori "piacevoli da avere" e si affidano a sforzi di "ricerca" o "giornata di hack".

- Miglioramenti dei processi o automazione possono essere introdotti mentre si affronta un nuovo requisito come una patch di sicurezza urgente, tuttavia non c'è supporto per costruire le soluzioni in modo riutilizzabile o sostenibile.
- I dipendenti si lamentano di burnout e frustrazione per la quantità di lavoro che stanno facendo al di fuori del loro ruolo principale.

Scenari Esempio:

- C'è un dipendente specifico che è considerato l'esperto dell'ambiente di test. Anche se questo dipendente ha buone intenzioni, il suo tentativo di abilitare ambienti di test migliori nonostante l'investimento limitato ha portato a un aumento del rischio poiché non c'è manutenzione della sua soluzione e nessuna comprensione condivisa di come gestire l'ambiente di test in caso di disservizio.
- Gli ingegneri sono incoraggiati ad investire in miglioramenti delle capacità quando non c'è pressione da parte della direzione su altre funzionalità legate più direttamente ai ricavi. Questo si traduce negli ultimi giorni di alcuni sprint in cui viene data priorità all'automazione e al miglioramento di parti della loro pipeline di CI/CD. Non è raro che questi miglioramenti arrivino a scatti, poiché possono esserci mesi di sprint eccessivamente pieni che non permettono tempo per queste iniziative secondarie.

Livello 2, Operativo - Team Dedicato

Il budget e le persone sono allocati per un supporto permanente di persone e risorse. Il personale assegnato ha il compito di fornire un insieme di capacità comunemente richieste per accelerare la consegna del software. Spesso questi team si concentrano sul soddisfare i requisiti tecnici reattivi. Possono essere chiamati DevOps, Engineering Enablement, Esperienza dello Sviluppatore (DevEx o DevX), Strumenti Condivisi, un Center of Excellence, o anche Piattaforma. Sono finanziati centralmente e trattati come centri di costo; il loro impatto sui flussi di valore diretti e sui team applicativi non è misurato. Può essere difficile mappare l'impatto dei team

di piattaforma a questo livello sull'organizzazione e sui suoi flussi di valore, il che può rendere difficile sostenere e continuare a finanziare tali team.

Caratteristiche:

- Il team è composto quasi interamente da generalisti tecnici.
- Il budget del team può includere i costi infrastrutturali associati al loro lavoro, portando spesso a essere un punto chiave nelle conversazioni sul budget.
- Gli elementi del backlog spaziano su numerose tecnologie, portando a frequenti e ampi cambi di contesto.
- Questo team è spesso il primo a colmare un vuoto che non viene ancora affrontato, anche se non nel campo dichiarato per il team. Questo team prende in carico risorse che non hanno un proprietario.
- Alle persone assegnate raramente viene dato il tempo o hanno l'esperienza con la ricerca sui clienti per validare i loro progetti o implementazioni.

Scenari Esempio:

- Gli sviluppatori di applicazioni sollevano un problema riguardante il lungo tempo di *build* delle loro applicazioni. A un team centralizzato viene assegnato il compito di ridurre il tempo di *build* del 50%. Risolvono questo problema raddoppiando la dimensione e la quantità dei runner CI, dato che non sono abbastanza vicini al software per migliorare individualmente il disegno delle applicazioni stesse. Questo crea una preoccupazione di budget per il loro team centralizzato poiché il guadagno di produttività non è direttamente misurabile rispetto all'aumento dei costi infrastrutturali.

Livello 3, Scalabile - Come Prodotto

L'investimento nelle piattaforme interne e nelle loro capacità è simile all'investimento nei prodotti in uscita di un'impresa e nei flussi di valore: basato sul valore che si prevede forniranno ai loro clienti. La gestione del prodotto e l'esperienza utente sono

considerate ed investite esplicitamente. Un sistema di addebito può essere utilizzato per riflettere l'impatto delle piattaforme sui propri flussi di valore diretti e prodotti dei clienti. L'impresa alloca fondi e personale alle iniziative appropriate utilizzando indicatori di performance basati sui dati e cicli di feedback. I team di piattaforma possono in ultima analisi ottimizzare l'azienda stessa e contribuire ad aumentare la redditività.

Caratteristiche:

- I ruoli del personale dei team di piattaforma non si trovano tradizionalmente in team interni o tecnici, ad esempio, gestione del prodotto ed esperienza utente.
- Il team pubblicizza internamente all'organizzazione una roadmap, che indica il valore consegnato e gli obiettivi delle funzionalità ad alto livello.
- Le funzionalità sono testate sia per la qualità dell'implementazione che per l'esperienza utente durante la progettazione, la consegna e il post-deployment.
- La rimozione delle funzionalità è una parte chiave della conversazione, l'obiettivo è avere una suite di capacità ben supportata e ben utilizzata invece di un vasto patrimonio che potrebbe non essere mantenuto.

Scenari Esempio:

- I dati derivati dalle metriche di utilizzo della piattaforma informano le decisioni per allocare fondi e personale alle iniziative più impattanti.

Livello 4, Ottimizzazione - Ecosistema Abilitato

I team di piattaforma trovano modi per aumentare l'efficienza e l'efficacia su tutta l'organizzazione oltre le capacità di base. I manutentori principali della piattaforma si sforzano intenzionalmente di ottimizzare il time-to-market per i nuovi prodotti, ridurre i costi su tutta l'impresa, abilitare una governance e una conformità efficienti

per i nuovi servizi, scalare i carichi di lavoro rapidamente e facilmente, e altri requisiti trasversali. Questi manutentori principali sono focalizzati su abilitare gli specialisti delle capacità a integrare con la massima fluidità e continuità i loro requisiti e offerte nelle parti esistenti e nuove delle piattaforme. Inoltre, l'organizzazione concentra persone e risorse da domini specialistici come sicurezza, performance, qualità nell'interagire con i framework della piattaforma forniti per introdurre funzionalità avanzate che possono permettere ai team di prodotto di accelerare la loro aderenza agli obiettivi aziendali senza dipendere da un backlog di team centralizzato.

Caratteristiche:

- Diventa una priorità abilitare gli specialisti ad estendere le capacità della piattaforma e introdurre di nuove.
- L'organizzazione può centralizzare gli specialisti, permettendo che la loro conoscenza e supporto si diffondano attraverso le capacità della piattaforma.

Scenari Esempio:

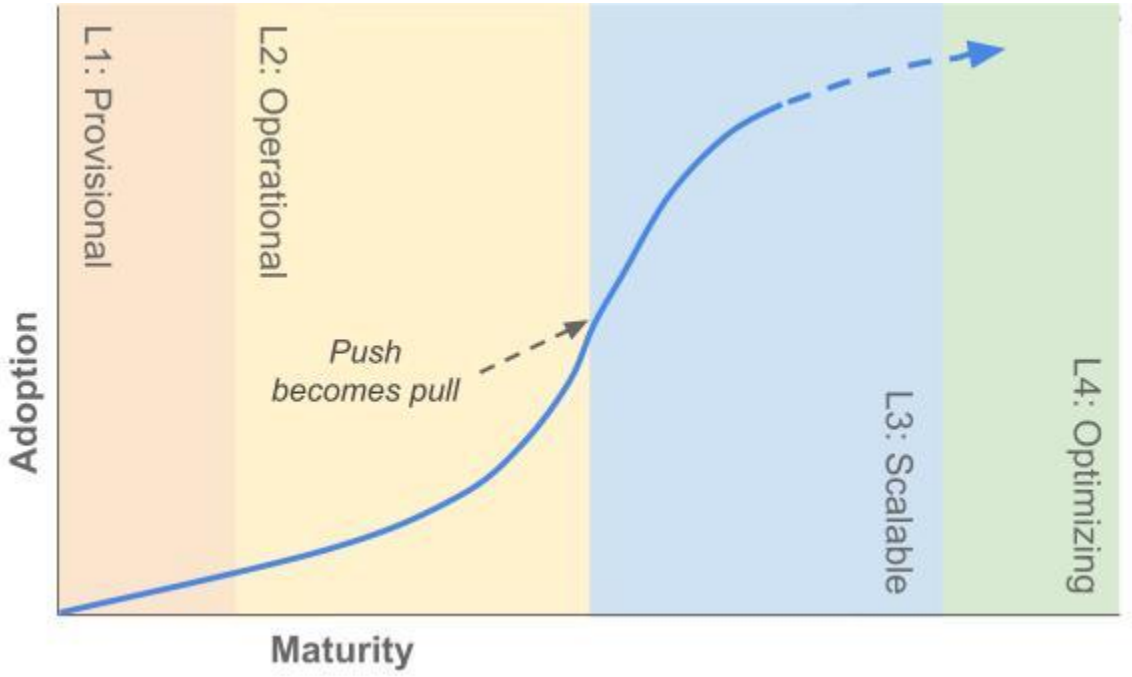
- Il marketing collabora con i costruttori della piattaforma per introdurre un tracciamento utente, coerente al fine di attribuire gli sforzi di marketing ai risultati del prodotto.
- L'iniziativa di automazione riduce il tempo umano necessario per il provisioning dei database di 30 minuti per istanza, risparmiando 10 milioni di dollari all'anno.

Adozione

Perché e come gli utenti scoprono e utilizzano le piattaforme interne e le capacità della piattaforma?

L'adozione descrive non solo come e quanto un'organizzazione utilizza le funzionalità della piattaforma, ma anche ciò che li motiva a farlo. Nelle fasi iniziali, molti utenti target potrebbero non rendersi conto di utilizzare una piattaforma, ma vedono i loro

strumenti come una raccolta ad hoc di funzionalità provenienti da varie fonti interne. Questo può maturare in un insieme di funzionalità che viene gestito e presentato agli utenti in modo coerente, cioè, una o più piattaforme. Man mano che le funzionalità diventano più raffinate e scopribili, è comune che la spinta all'uso della piattaforma si allontani da motivazioni più esterne come mandati o incentivi. Questo porta gli utenti a “auto-selezionarsi” nelle funzionalità della piattaforma e, idealmente, anche a investire i propri sforzi nell'ecosistema più ampio della piattaforma.



Un diagramma per indicare un modello di crescita comune per l'adozione della piattaforma. Questo mette in mostra l'avvio spesso lento guidato principalmente dai costruttori della piattaforma. Una volta che le piattaforme forniscono abbastanza valore agli utenti, la crescita diventa maggiormente trascinata dagli utenti, causando una curva di adozione più ripida.

Livello 1, Provvisorio - Erratica

L'adozione di piattaforme condivise e funzionalità è sporadica e inconsistente. Non esiste una strategia o una guida su scala aziendale per scegliere e integrare i servizi di supporto e le tecnologie richieste. I singoli team potrebbero sfruttare le pratiche di piattaforma per migliorare i propri processi, ma non c'è uno sforzo coordinato o una standardizzazione in tutta l'organizzazione. Questo livello di adozione è caratterizzato

dall'assenza di un approccio coerente e dall'idea che gli strumenti esterni siano più efficaci di quelli forniti internamente.

Caratteristiche:

- Strumenti, servizi e funzionalità *una tantum* sono gestiti e consumati da vari team e dipartimenti nell'organizzazione.
- I servizi gestiti da chi fornisce l'nfrastruttura (tipicamente noto come "cloud provider") sono adottati e utilizzati in modo incoerente e senza pratiche e politiche standard, poiché le configurazioni interne sono difficili da trovare o utilizzare.
- I team di app e servizi scoprono strumenti e capacità casualmente, tramite voci di corridoio e conversazioni fortuite piuttosto che attraverso un processo più centralizzato.
- Il coordinamento e il riutilizzo di componenti e capacità sono guidati solo dagli utenti finali (team di sviluppo), se non del tutto assenti.
- Ogni team di prodotto mantiene il proprio insieme di script o strumenti per distribuire le loro applicazioni.

Scenari Esempio:

- Un servizio bancario richiede un database. Uno sviluppatore scopre da un amico di un altro team che possono richiedere un account AWS e configurare un database RDS. Da un altro team trovano uno script Terraform per provisionare quel database. Per il monitoraggio usano CloudWatch su base ad hoc; copiano manualmente i secrets dalla console AWS in un'istanza di HashiCorp Vault prima di eseguire lo script Terraform.

Livello 2, Operativo — Spinta Estrinseca

L'organizzazione riconosce il valore delle piattaforme e delle capacità condivise e si sforza di incoraggiare e coltivarle. Direttive interne incentivano o addirittura

richiedono l'uso di servizi di piattaforma condivisa per alcuni casi d'uso. Alcuni team di prodotto utilizzano le funzionalità della piattaforma più di altri; le funzionalità coprono i casi d'uso tipici nell'organizzazione ma non quelli insoliti; ed è difficile aggiungere queste eccezioni alla piattaforma comune.

La scoperta delle funzionalità da parte degli utenti e come utilizzarle è inconsistente; è possibile che un utente di un team di prodotto non scopra una funzionalità supportata a meno che non venga indirizzato da un team di piattaforma.

Caratteristiche:

- Un certo grado di impulso esterno porta all'uso delle funzionalità della piattaforma, ad esempio:
    - Incentivi come le recensioni personali
    - Mandati come l'obbligo di utilizzo per i rilasci in produzione o la ricezione di finanziamenti
- L'utilizzo delle funzionalità della piattaforma è frammentato: gli utenti possono sfruttare una funzionalità ma potrebbero non essere a conoscenza di, o interessati ad adottare, altre che sono disponibili.
- Gli utenti hanno una bassa motivazione ad imparare come utilizzare le funzionalità della piattaforma e si affidano fortemente alla collaborazione con i fornitori attraverso forum come orari di ricevimento o help desk.
- Gli utenti della piattaforma sono incoraggiati a in comunità di pratica informali per condividere problemi e soluzioni, ma la partecipazione può essere limitata.

Scenari Esempio:

- Un'organizzazione di ingegneria decide su uno strumento di distribuzione standard e istruisce tutti i team all'utilizzo. Nuovi processi (comunicazione delle note di rilascio, ecc.) sono costruiti attorno a quello standard. Ai team è ordinato di smettere di usare altri tipi di script di distribuzione e di usare invece lo strumento comune. Questo è difficile per alcuni team le cui esigenze non

sono soddisfatte dal nuovo processo ma che non capiscono o non sono autorizzati ad estenderlo.

Livello 3, Scalabile - Trazione Intrinseca

Gli utenti dei team di prodotto e servizio scelgono di utilizzare le piattaforme e le loro capacità a causa del chiaro valore che forniscono nel ridurre il carico cognitivo sui team di prodotto, fornendo al contempo servizi di supporto di superiore qualità. Documentazione e interfacce ergonomiche permettono agli utenti del team di prodotto di provvedere rapidamente e utilizzare le funzionalità della piattaforma. Gli utenti scelgono implementazioni di piattaforme interne rispetto ad alternative, come sviluppare la funzionalità da soli o assumere un fornitore.

Caratteristiche:

- L'adozione della piattaforma è autosufficiente: il principale motore d'adozione non è un impulso esterno o un incentivo che obbliga gli utenti a utilizzare le offerte della piattaforma - piuttosto, sono i valori di queste offerte di piattaforma stessi che attirano gli utenti.
- Gli utenti valutano ed apprezzano una consistenza di funzionalità, fornite dalla piattaforma. C'è l'aspettativa che una funzionalità individuale non sia isolata, ma che sia una caratteristica tra un insieme più grande di caratteristiche della piattaforma.
- I team di piattaforma incoraggiano l'adozione naturale delle piattaforme raccogliendo feedback degli utenti, condividendo una roadmap e mantenendo forum aperti per conversazioni con gli utenti.
- I team di sviluppo e prodotto apprezzano abbastanza le funzionalità della piattaforma da pagarle, ad esempio, tramite un sistema di addebito.
- Gli utenti possono condividere feedback e conoscere le funzionalità in arrivo tramite forum aperti e roadmap condivise.
- Portali self-service, template per golden-path e altri documenti consentono un uso rapido.

Scenari Esempio:

- Un team di sviluppo ha avuto successo nella richiesta di un nuovo database. Il loro processo era facile da comprendere e richiedeva un tempo trascurabile o nullo di attesa. Inoltre, funzionalità chiave come i backup e il monitoraggio, che hanno permesso al team di procedere con l'utilizzo fino alla produzione senza problemi, erano incluse. Questa esperienza significa che, quando il team, in seguito, aveva nuovamente bisogno di una risorsa, il loro primo istinto è stato quello di optare per la piattaforma interna. Anche se inizialmente intendevano usare una specifica tecnologia, alla fine hanno scelto di utilizzare quella offerta internamente, poiché sapevano quanto bene la soluzione sarebbe stata integrata per la loro organizzazione.

Livello 4, Ottimizzazione - Partecipativa

Gli utenti dei team di prodotto investono ulteriormente nelle capacità della piattaforma, unendosi all'ecosistema e contribuendo ad esso. Alcuni contributi migliorano e correggono le funzionalità esistenti; altri introducono nuove capacità e funzionalità per affrontare nuovi casi d'uso. Processi e servizi sono definiti e abilitano gli utenti a identificare requisiti e coordinare contributi tra diversi team di prodotto e piattaforma. Nuove capacità sono pubblicate tramite interfacce e portali in modo consistente e con documentazione completa e versionamento standard.

Caratteristiche:

- Gli utenti nei team di app/servizio sono autorizzati a contribuire con correzioni, funzionalità e feedback per le capacità della piattaforma.
- Progetti esterni e standard sono strategicamente sfruttati per ridurre i costi di manutenzione, accelerare la consegna di nuove funzionalità e utilizzare il numero di dipendenti dell'organizzazione nel modo più efficace.

- Nuove funzionalità e miglioramenti sono coordinati asincronicamente attraverso issue boards e pull request. Documenti e checklist abilitano lo sviluppo autonomo da parte dei contributori.
- I Developer Advocate e gli ambasciatori interni costruiscono e supportano una comunità di utenti interni che estende la proprietà della piattaforma anche ai contributori dei team di app e servizio.
- L'uso delle capacità della piattaforma è visto come il miglior modo di lavorare nell'organizzazione sia dalla leadership che dai singoli contributori.
- Gli ingegneri della piattaforma partecipano alla pianificazione del team di prodotto per apprendere i requisiti e suggerire capacità esistenti rilevanti.

Scenari Esempio

- Un team desidera un piano di backup alternativo. Dopo aver proposto ciò come offerta generale, la richiesta viene classificata di bassa priorità a causa del minimo riutilizzo. Il team richiedente sceglie di integrare la propria soluzione nel framework della piattaforma e renderla disponibile all'organizzazione. Inizialmente è un'offerta *alpha*, ma una volta soddisfatti tutti i requisiti operativi può essere promossa a capacità fondamentale della piattaforma.

Interfacce

Come interagiscono e consumano gli utenti le funzionalità della piattaforma?

Le interfacce fornite dalle piattaforme influenzano il modo in cui gli utenti interagiscono con queste offerte di piattaforma per fornire, gestire e osservare le funzionalità. Le interfacce possono includere sistemi di ticketing, template di progetto e portali grafici così come API automatizzabili e strumenti da linea di comando (CLI).

Caratteristiche chiave di un'interfaccia includono quanto è scopribile e facile da usare durante i principali percorsi utente come la richiesta iniziale, la manutenzione o il triage degli incidenti. Livelli di maturità più elevati qui riflettono interfacce più integrate, consistenti, automatizzate e supportate.

Livello 1, Provvisorio — Processi Personalizzati

Esiste una raccolta di processi per fornire diverse funzionalità e servizi, ma la coerenza dell'interfaccia non è considerata. Processi su misura soddisfano le necessità immediate di individui o team e si affidano all'intervento manuale anche se il fornitore utilizza alcuni script di implementazione automatizzati.

La conoscenza di come richiedere queste soluzioni è condivisa da persona a persona. Il processo per richiedere un servizio manca di standardizzazione e coerenza. Fornire e utilizzare un servizio di piattaforma richiede probabilmente un profondo supporto da parte del fornitore della capacità.

La mancanza di requisiti e standard centralizzati rende questo livello appropriato quando l'azienda non ha ancora identificato e documentato le aspettative. Può essere particolarmente efficace per team in aziende in fase iniziale o sforzi di piattaforma. In questi ambienti, ai team viene fornita la libertà di evolvere processi e capacità secondo le loro necessità, permettendo loro di consegnare più

rapidamente e pagare il prezzo della standardizzazione solo quando necessario in seguito.

Caratteristiche:

- L'interazione con l'utente non è un argomento chiave di discussione e raramente (se mai) le interazioni vengono testate durante la progettazione e la consegna di nuove funzionalità.
- Le funzionalità sono principalmente fornite tramite richieste manuali, anche se i fornitori possono scegliere di automatizzare alcune o tutte le attività necessarie per soddisfare una richiesta dell'utente.
- Le richieste che apparentemente sono "semplici" diventano complesse a causa della necessità di scoprire il processo giusto da seguire.
- A volte un processo sembra essere approvato, ma gli utenti incontrano problemi quando si coinvolge un dipartimento o team diverso.

Scenari Esempio:

- Un team di applicazione vuole testare le prestazioni della loro nuova modifica. Per fare ciò, vogliono un ambiente isolato che contenga abbastanza dati di test per ottenere una lettura accurata delle prestazioni. L'ultima volta che hanno avuto questa richiesta, un ex compagno di team era riuscito ad accedere a un ambiente, ma da allora se n'è andato e nessuno sa come ricrearlo. Alla fine, vengono messi in contatto con un ingegnere del team di infrastruttura che è in grado di fornire loro un ambiente in pochi giorni.
- Un team nelle fasi esplorative dello sviluppo di un prodotto utilizza un processo su misura per fornire un nuovo servizio cloud senza dover convalidare che la loro soluzione giustifichi ulteriori investimenti.

Livello 2, Operativo - Strumentazione Standard

Esistono interfacce standard e coerenti per la fornitura e l'osservazione di piattaforme e funzionalità che soddisfano esigenze ampie. Gli utenti sono in grado di identificare quali funzionalità sono disponibili e sono abilitati a richiedere le funzionalità di cui hanno bisogno.

"Strade lastricate" o *"golden paths"*, sotto forma di documentazione e modelli, sono forniti. Queste risorse definiscono come fornire e gestire le funzionalità tipiche utilizzando schemi conformi e testati. Sebbene alcuni utenti siano in grado di utilizzare queste soluzioni autonomamente, le soluzioni richiedono spesso ancora un'approfondita competenza di dominio e quindi il supporto dei manutentori è ancora vitale.

Caratteristiche:

- Le soluzioni tecniche sono costruite in strumenti specifici per il loro dominio di problema, non sempre strumenti familiari agli utenti.
- C'è un investimento in un percorso comune; tuttavia, deviare da quel percorso rivela rapidamente poche opzioni di personalizzazione poiché l'attenzione era sulla costruzione di un'unica opzione.
- Data la standardizzazione, gruppi interni informali sono in grado di formarsi e riunirsi per condividere buone pratiche e superare problemi condivisi.
- Potrebbe esserci una divergenza nell'implementazione delle funzionalità poiché i team prendono modelli, li personalizzano e poi non possono integrare modifiche dal team centralizzato.

Scenari Esempio:

- Un team centralizzato cura una libreria di moduli Terraform, controller Kubernetes e CRD per la fornitura di diversi tipi di infrastruttura.
- Una location condivisa include documenti completi sulle soluzioni in tutta l'organizzazione.

Livello 3, Scalabile - Soluzioni Self-Service

Le soluzioni sono offerte in modo da fornire autonomia agli utenti e richiedere poco supporto dai manutentori. L'organizzazione incoraggia e abilita le soluzioni a fornire interfacce coerenti che permettono la rilevabilità e la portabilità dell'esperienza utente da una funzionalità all'altra. Sebbene self-service, le soluzioni richiedono consapevolezza e implementazione del team. Per migliorare questa esperienza potrebbe esserci un linguaggio interno guidato e semplificato che permette agli utenti di adottare e integrare le funzionalità della piattaforma più rapidamente. Questo genera una raccolta di funzionalità centrata sull'utente, self-service e coerente.

Caratteristiche:

- Le soluzioni sono fornite come implementazioni "con un clic", consentendo ai team di beneficiare di una determinata funzionalità senza dover capire come venga fornita.
- Sebbene le soluzioni siano facili da creare, potrebbe non esserci tanta usabilità incorporata nella gestione della soluzione dal day-2 in poi.
- Continua ad esserci un percorso ristretto di soluzioni disponibili, lasciando gli utenti con requisiti unici incerti su come procedere.

Scenari Esempio:

- Viene fornita un'API che astrae la creazione e la manutenzione di database e fornisce agli utenti tutte le informazioni di cui hanno bisogno per sfruttare quella capacità della piattaforma come una stringa di connessione, la posizione per i dati segreti e una dashboard con dati di osservabilità.

Livello 4, Ottimizzazione - Servizi Gestiti

Le funzionalità della piattaforma sono integrate in modo trasparente negli strumenti e nei processi che i team utilizzano già per svolgere il loro lavoro. Alcune funzionalità sono fornite automaticamente, come l'osservabilità o la gestione dell'identità per un servizio distribuito. Quando gli utenti raggiungono i limiti dei servizi forniti, c'è l'opportunità di superare le soluzioni automatizzate e personalizzare in base alle loro esigenze senza abbandonare le offerte interne perché le capacità della piattaforma sono considerate blocchi costruttivi. Questi blocchi costruttivi sono utilizzati per costruire composizioni trasparenti e automatiche per soddisfare i casi d'uso di livello superiore consentendo allo stesso tempo una personalizzazione più profonda dove necessario.

Caratteristiche:

- È chiaro quali capacità differenziano l'organizzazione e quali no, consentendo ai team interni di investire in soluzioni personalizzate solo dove non possono sfruttare gli standard dell'industria.
- Sebbene le funzionalità siano presentate in modo coerente, non esiste un unico modo di utilizzare le funzionalità. Alcune sono più adatte come strumenti CLI da utilizzare negli script, mentre altre traggono vantaggio dall'integrazione in editor o IDE dove l'utente scrive e sviluppa codice.
- Il valore delle singole funzionalità è ampliato con un focus sul flusso dello sviluppo e del rilascio del software, portando a un focus su come combinare le funzionalità in offerte di livello superiore.
- Sebbene le funzionalità siano spesso fornite in pacchetti, i super utenti sono abilitati a decomporre queste offerte di livello superiore al fine di ottimizzare quando e dove ne hanno bisogno.

Scenari Esempio:

- Gli agenti di osservabilità sono agganciati ad ogni carico di lavoro e un proxy OIDC è posizionato davanti a tutte le applicazioni.
- Per impostazione predefinita, ogni nuovo progetto riceve uno spazio in un task runner (pipeline) e un ambiente di runtime (namespace Kubernetes), tuttavia un progetto può scegliere altre opzioni come il runtime serverless.
- Da un catalogo in un portale Service Now un utente seleziona "Fornisci un Database". L'automazione fornisce un database RDS e invia all'utente un URL e la posizione per ottenere le credenziali.

Operatività

Come vengono pianificate, priorizzate, sviluppate e mantenute le piattaforme e le loro funzionalità?

Garantire l'operatività di una piattaforma significa gestire e supportare le sue capacità e funzionalità per tutta la loro durata, inclusa l'accettazione di nuove richieste, i rilasci iniziali, gli aggiornamenti e le estensioni, la manutenzione e le operazioni continue, il supporto agli utenti, e persino la deprecazione e la terminazione. Le organizzazioni e i loro team di piattaforma scelgono piattaforme e funzionalità da creare e mantenere e possono dare priorità alle iniziative più preziose e impattanti.

È da notare che la maggior parte del lavoro per fornire una funzionalità viene speso dopo il suo rilascio iniziale - nel fornire aggiornamenti senza interruzioni, nuove funzionalità migliorate, supporto operativo e abilitazione ed educazione degli utenti finali. Pertanto, un platform engineering team impattante pianificherà in anticipo e gestirà la piattaforma in modo adeguato a garantire operazioni sostenibili a lungo termine e affidabilità in generale.

Livello 1, Provvisorio — Su richiesta

Piattaforme e funzionalità sono sviluppate, pubblicate e aggiornate in modo reattivo, in base a richieste e requisiti ad hoc dei team di prodotto. I team di prodotto stessi potrebbero addirittura dover pianificare e costruire le funzionalità di cui necessitano.

I team che sviluppano una nuova funzionalità, sia che si tratti di team centralizzati dedicati sia di team di applicazione che soddisfano le proprie esigenze, assumono solo una responsabilità informale nel supportare gli altri nell'uso. Non si prevede che la mantengano attivamente ed esistono pochi processi per verificare la qualità dell'offerta. A questo livello, le implementazioni sono spesso ignorate fino alla scoperta di una vulnerabilità di sicurezza, fino a che un bug non ne impedisce l'uso, o fino all'arrivo di un nuovo requisito, momento in cui può essere rapidamente implementato un altro piano reattivo.

Caratteristiche:

- Le funzionalità sono create per soddisfare le esigenze urgenti dei singoli team di applicazione.
- L'attenzione è rivolta alla consegna iniziale delle funzionalità fondamentali; non si effettuano piani per la manutenzione continua e la sostenibilità.
- Le implementazioni delle funzionalità sono generalmente datate e in attesa di aggiornamenti.
- Picchi improvvisi di lavoro sono introdotti per cambiamenti di grande impatto alle funzionalità, come la scoperta di una vulnerabilità.
- I cambiamenti possono risultare in downtime sia pianificati che imprevisti.
- Ogni aggiornamento viene effettuato in modo personalizzato, richiedendo tempo e ricerca per ideare un processo ad ogni aggiornamento.

Scenari Esempio:

- Viene annunciata la vulnerabilità di sicurezza Log4Shell e l'organizzazione costituisce un team specializzato per investigare dove possa essere vulnerabile e avviare le patch. Una volta che il team identifica l'impatto, deve collaborare strettamente con numerosi team diversi, dato che ognuno gestisce i propri server e i propri processi di aggiornamento in modo differente. Anche quando questo lavoro viene considerato completo, il livello di fiducia è relativamente basso, al punto che che non emergeranno ulteriori istanze di miglioramento.

Livello 2, Operativizzato — Tracciato centralmente

Piattaforme e funzionalità sono documentate e scopribili centralmente, e i processi per la pianificazione e la gestione del ciclo di vita delle funzionalità sono almeno leggermente definiti. La responsabilità e la proprietà sono documentate per ciascun servizio e funzione. I processi di gestione del ciclo di vita variano per diverse capacità a seconda dei loro proprietari e delle loro priorità. Un team centralizzato mantiene, o è in grado di generare su richiesta, un inventario delle capacità nel backlog per fornire lo stato della manutenzione per le funzionalità correnti. Ciò consente all'organizzazione di tracciare i progressi verso l'offerta di funzionalità e la conformità con i requisiti di aggiornamento.

Caratteristiche:

- I team di applicazione creano nuove funzionalità secondo necessità per soddisfare esigenze urgenti.
- Un team centrale fornisce un registro dei servizi condivisi disponibili in tutta l'organizzazione.
- Standard flessibili, come la richiesta di un'API automatizzabile e documentazione sull'uso, sono applicati alle funzionalità.

- L'Infrastruttura come Codice è utilizzata per consentire una più facile tracciabilità dei servizi distribuiti.
- Audit per regolamenti di conformità come PCI DSS o HIPAA sono abilitati tramite gli inventari dei servizi.
- Il lavoro di migrazione e aggiornamento è tracciato contro un grafico di esaurimento che consente all'organizzazione di tracciare il tasso di conformità e il tempo fino al completamento.
- Il tracciamento non indica il livello di supporto; spesso gli aggiornamenti in questa fase sono ancora manuali e personalizzati.

Scenari Esempio:

PostgreSQL 11 sta andando EOL entro la fine dell'anno. L'organizzazione è a conoscenza di quali database richiedono l'aggiornamento e sta pianificando il lavoro nel backlog di ciascun team per completarlo.

Livello 3, Scalabile — Abilitato centralmente

Le piattaforme e le funzionalità non sono solo registrate centralmente ma anche orchestrate centralmente. I team di piattaforma assumono la responsabilità di comprendere le esigenze ampie dell'organizzazione e di conseguenza di dare priorità al lavoro attraverso i team di piattaforma e di infrastruttura. Coloro che sono responsabili di una funzionalità sono attesi non solo a mantenerla tecnicamente, ma anche a fornire esperienze utente standard per integrare la funzionalità con altri servizi correlati in tutta l'organizzazione, garantire un uso sicuro e affidabile e persino fornire osservabilità.

Esistono processi standard per creare ed evolvere nuove funzionalità, consentendo a chiunque nell'organizzazione di contribuire con una soluzione che soddisfi le aspettative. I processi di consegna continua per le capacità e le funzionalità della piattaforma consentono il rollout regolare e il rollback. Grandi cambiamenti sono

pianificati e coordinati come lo sarebbero per i cambiamenti dei prodotti rivolti ai clienti.

Caratteristiche:

- I team di applicazione richiedono i servizi dai team di piattaforma prima di crearli.
- I nuovi servizi devono aderire a pratiche standard come interfacce standard, documentazione e governance.
- I processi di aggiornamento sono documentati e consistenti attraverso versioni e servizi.
- Dove il fornitore della funzionalità non gestisce un aggiornamento, forniscono strumentazione e supporto agli utenti per un impatto minimo.

Scenari Esempio:

- L'organizzazione sta per passare a RHEL 9. In tal caso, ogni team di applicazione deve validare che il proprio software continui a funzionare. Per abilitare questa fase di test, il team di calcolo centralizzato sta configurando ambienti di test per ogni team con le corrette versioni di software e sistema operativo.

Livello 4, Ottimizzazione — Servizi gestiti

Il ciclo di vita di ogni funzionalità è gestito in modo standardizzato e automatizzato. Capacità, funzionalità e aggiornamenti sono consegnati continuamente senza impatto sugli utenti. Qualsiasi grande cambiamento iniziato dai fornitori di piattaforma include piani di migrazione per gli utenti esistenti con responsabilità e tempistiche definite.

I fornitori di funzionalità della piattaforma assumono la maggior parte della responsabilità per la manutenzione, ma esiste un contratto chiaro - un "modello di responsabilità condivisa" - che descrive le responsabilità degli utenti, consentendo ad entrambe le parti di operare per lo più autonomamente.

Caratteristiche:

* Un modello di proprietà condivisa definisce chiaramente chi è responsabile delle piattaforme e delle loro funzionalità e cosa si aspetta dagli utenti.
* I team scriptano sia l'esecuzione dell'aggiornamento sia eventuali strategie di rollback per mantenere bassi il rischio e l'impatto.

Scenari Esempio:

- Gli utenti di macchine virtuali non sono tenuti a gestire nulla riguardo gli aggiornamenti di versione. Il loro unico requisito è avere una fase nel loro pipeline di consegna che contenga un test del fumo rappresentativo. Viene poi chiesto loro di dichiarare la loro applicazione come avente una tolleranza al rischio inferiore per aspettare un aggiornamento completamente consolidato o una tolleranza maggiore per diventare un utilizzatore precoce. La capacità della macchina virtuale gestisce quindi il rilascio automatizzato degli aggiornamenti, inclusi i rollback dopo fallimenti dello *smoke test* o del rilascio *canary*.

Misurazione

Qual è il processo per raccogliere e incorporare feedback e apprendimenti?

Rispondendo ai feedback espliciti e impliciti degli utenti, le organizzazioni possono aumentare la soddisfazione degli utenti e garantire la sostenibilità a lungo termine della piattaforma. Le organizzazioni devono bilanciare l'innovazione e il soddisfare le richieste degli utenti per mantenere la rilevanza della piattaforma. Man mano che la tecnologia e le preferenze degli utenti cambiano, le piattaforme più agili e reattive a questi cambiamenti avranno maggiore successo. Rivisitare e affinare regolarmente il meccanismo di feedback può ulteriormente ottimizzare lo sviluppo della piattaforma e migliorare l'engagement degli utenti.

Livello 1, Provvisorio — Ad hoc

Le metriche di uso e soddisfazione vengono raccolte in modi personalizzati, se è il caso, per ogni piattaforma e funzionalità. I risultati e le metriche di successo non sono allineati in modo coerente tra le funzionalità e, di conseguenza, le intuizioni corrispondenti non vengono raccolte. Il feedback degli utenti e la strumentazione dell'uso della piattaforma potrebbero non essere raccolti, o se lo sono, sarà in modo informale. Le decisioni vengono prese basandosi su requisiti aneddotici e dati incompleti.

Caratteristiche:

- Nessuna esperienza o opinioni su come misurare il successo delle piattaforme
- Utilizzo di strumenti familiari per raccogliere metriche comuni, con intento e premeditazione limitati
- Dipendenza da piccole quantità di dati
- Difficoltà a garantire la partecipazione degli utenti — gli utenti credono che il loro feedback non venga considerato

- Se vengono utilizzati sondaggi, le domande cambiano tra una sessione e l'altra, annullando la capacità di tracciare i progressi

Scenari Esempio:

- Un responsabile tecnico della piattaforma vuole migliorare la collaborazione con gli utenti aggiungendo argomenti chiave al loro prossimo piano trimestrale. Decide di eseguire un sondaggio su cosa gli utenti vorrebbero vedere. Il tasso di risposta è molto alto, il che è eccitante, ma risulta anche difficile organizzare e rispondere a tutte le idee. Anche se alcune idee influenzano la pianificazione trimestrale, gli utenti non vedono le loro idee come accettate e sono meno inclini a rispondere al prossimo sondaggio.
- Il team vuole catturare più dati automaticamente, quindi cerca opportunità per una facile raccolta, come i fallimenti dei test in CI. Tuttavia, non tutti i team utilizzano la stessa automazione CI quindi i dati sono disponibili solo per le applicazioni Java, anche se alcuni team sono passati a scrivere i loro servizi in Scala.

Livello 2, Operativo — Raccolta coerente

Le organizzazioni a questo livello hanno l'obiettivo intenzionale di verificare che i prodotti della piattaforma soddisfino le esigenze del loro mercato di utenti interni. Viene valorizzata una raccolta strutturata e azionabile del feedback degli utenti. Potrebbero essere assegnati team dedicati o singoli individui per raccogliere il feedback, garantendo un approccio più coerente. I canali di feedback, come sondaggi o forum degli utenti, sono standardizzati e il feedback è categorizzato e prioritizzato. Oltre al feedback degli utenti, c'è anche l'aspettativa che le esperienze degli utenti siano valorizzate per generare dati di utilizzo nel tempo.

Rimangono sfide nel tradurre il feedback in compiti azionabili. Mentre c'è un crescente repository di dati degli utenti, l'organizzazione potrebbe avere difficoltà a

comprendere ed integrare efficacemente questo feedback in una roadmap della piattaforma. Può essere difficile garantire che gli utenti vedano cambiamenti tangibili guidati dal loro feedback.

Caratteristiche:

- La raccolta dei dati è discussa come argomento della maggior parte delle sessioni di pianificazione principali o delle implementazioni delle funzionalità.
- Potrebbe non esserci allineamento su cosa esattamente misurare per verificare il successo.
- Le funzionalità della piattaforma possono essere misurate per il successo, ad esempio misurando l'adozione degli utenti o il tempo risparmiato dagli utenti.

Scenari Esempio:

- Un team della piattaforma dedica il 20% del loro tempo a funzionalità definite dagli utenti, che identificano in base a sondaggi e altre tecniche di intervista. Le loro scoperte sono raccolte in uno strumento che consente ulteriori votazioni e commenti per affinare ulteriormente le priorità. Durante l'implementazione, gli utenti che hanno fatto la richiesta sono avvicinati per collaborare sui primi disegni e implementazioni. Una volta implementate, vengono fatti annunci che assicurano che gli utenti richiedenti siano a conoscenza delle nuove funzionalità e supportati nell'adozione.
- Il team focalizzato sulle capacità di consegna del software vuole catturare più dati automaticamente, inclusi il ciclo di tempo che automatizzano attraverso lo strumento di build dal commit alla produzione. C'è la comprensione che il ciclo di tempo può includere altre attività come la revisione PR, ma questo al momento non è incluso.

Livello 3, Scalabile — Intuizioni

Sebbene esistano già robusti meccanismi di feedback standard, a questo stadio i dati vengono raccolti in modi studiati per produrre specifiche intuizioni strategiche e azioni. I risultati e gli esiti desiderati vengono identificati e poi vengono scelte metriche standard per indicare il progresso verso quegli esiti. Framework e standard dell'industria possono essere utilizzati per beneficiare della ricerca dell'industria sull'impatto di certi comportamenti.

Team dedicati o strumenti sono impiegati per raccogliere e rivedere il feedback e riassumere intuizioni azionabili. Viene stabilita una relazione simbiotica tra i prodotti della piattaforma e i loro utenti. Il feedback è considerato un asset strategico che guida le operazioni della piattaforma e la roadmap. Potrebbero essere istituite regolari sessioni di revisione del feedback, dove team interfunzionali si riuniscono per discutere e strategizzare basandosi su intuizioni degli utenti.

Caratteristiche:

- Prima di consegnare qualsiasi nuova funzionalità della piattaforma, il team discute su come valutare l'esito del loro lavoro.
- L'organizzazione ha un allineamento ampio sulle misure che indicano il successo delle iniziative della piattaforma.
- Un product manager o un membro dedicato del team guida un processo di raccolta e analisi del feedback continuo e coerente.
- L'organizzazione ha stabilito metriche e obiettivi da osservare e puntare per indicare il successo.

Scenari Esempio:

- L'organizzazione ha tracciato costantemente i tempi di build e il lead time. Tuttavia, ora si rendono conto che, sebbene facili da raccogliere, questi da soli non forniscono un quadro completo della consegna del software. Con questo in mente, il team implementa la misurazione per l'affidabilità e la stabilità del servizio.

Livello 3, Scalabile — Insights

Sebbene esistano già meccanismi di feedback robusti e standardizzati, a questo stadio i dati sono raccolti in modi articolati per produrre intuizioni strategiche e azioni specifiche. I risultati e gli esiti desiderati sono identificati e tracciati conmetriche standard scelte per indicare il progresso verso tali esiti. Framework e standard del settore possono essere utilizzati per beneficiare della ricerca industriale sull'impatto di determinati comportamenti.

Team dedicati o strumenti sono impiegati per raccogliere e rivedere il feedback e riassumerli in spunti di comprensione azionabili. Si stabilisce una relazione simbiotica tra i prodotti della piattaforma e i loro utenti. Il feedback è considerato un bene strategico che guida le operazioni della piattaforma e la roadmap. Potrebbero essere istituite sessioni regolari di revisione del feedback, dove team interfunzionali si riuniscono per discutere e pianificare in base alle informazioni fornite dagli utenti.

Caratteristiche:

- Prima di consegnare qualsiasi nuova funzionalità della piattaforma, il team discute su come valutare l'esito del loro lavoro.
- L'organizzazione ha un ampio allineamento sulle misure che indicano il successo delle iniziative della piattaforma.
- Un Product Manager o un membro del team dedicato guida un processo continuo e coerente di raccolta e analisi del feedback.
- L'organizzazione ha stabilito metriche e obiettivi per osservare e puntare a indicare il successo.

Scenari Esempio:

- L'organizzazione ha tracciato costantemente i tempi di build e il lead time. Tuttavia, ora si rende conto che, sebbene facili da raccogliere, questi da soli non offrono un quadro completo della capacità di consegna del software. Con questo in mente, il team implementa una nuova misurazione per l'affidabilità e la stabilità del servizio.

Livello 4, Ottimizzazione — Quantitativo e qualitativo

Feedback e misurazioni sono profondamente integrati nella cultura dell'organizzazione. L'intera organizzazione su scala aziendale, dagli *executive* di alto livello agli ingegneri, riconosce il valore della raccolta di dati e del feedback sull'evoluzione del prodotto. C'è una democratizzazione dei dati, dove vari stakeholder, inclusi gli utenti della piattaforma e i leader aziendali, sono attivamente coinvolti nell'identificare ipotesi per miglioramenti della piattaforma, fornire feedback durante il processo di design e poi misurare l'impatto dopo la consegna. Tutte queste misurazioni sono considerate nella pianificazione delle iniziative della piattaforma.

Non solo vengono sfruttati framework standard, ma c'è la comprensione che misurare da più angolazioni crea un quadro più olistico. C'è un investimento nella comprensione di come le misure qualitative cambiano man mano che quelle quantitative migliorano. C'è un focus sull'identificazione di misure prospettiche che possono permettere l'anticipazione di funzionalità che sopporterebbero le esigenze degli utenti, allevierebbero le loro sfide e consentirebbero di restare avanti rispetto alle tendenze del settore e ai requisiti aziendali.

Caratteristiche:

- I team di piattaforma cercano continuamente modi per migliorare le metriche che osservano e il modo in cui raccolgono dati.
- L'organizzazione è familiare e sensibile alla *Legge di Goodhart*(: "Quando una misura diventa un obiettivo, smette di essere una buona misura."
- Le metriche e la telemetria raccolte sono continuamente valutate per un vero insight e valore.
- La gestione dei dati metrici è ben supportata, come le capacità standard della piattaforma per gestire data lakes e derivarne insight.
- È incoraggiata la collaborazione interdipartimentale per evitare silos di dati e abilitare cicli di feedback efficaci.

Scenari Esempio:

- Nel tempo, l'organizzazione ha raccolto dati che indicano un aumento del tempo di build di oltre il 15%. Questo innesca esperienze negative per gli sviluppatori e, una volta innescato, anche se il tempo di build viene ridotto sotto il tempo originale, gli sviluppatori restano frustrati più a lungo. Questa intuizione spinge il team di build a stabilire e aderire a un Obiettivo di Livello di Servizio (SLO), che consente l'identificazione precoce e il miglioramento prima di innescare il ciclo negativo con i loro utenti.

Conclusioni

Le piattaforme e i loro manutentori forniscono una base per lo sviluppo agile di prodotti digitali. Offrono una raccolta coerente di capacità e funzionalità che abilitano lo sviluppo e il rilascio efficienti del software. Questo modello di maturità fornisce una mappa per il vostro viaggio nel Platform Engineering.

Nota di traduzione

Per una migliore comprensione del contesto e al fine di evitare ambiguità, occasionalmente alcuni termini specifici — come *platform engineering*, *developer advocate*, *build*, *smoke test*, … — sono stati mantenuti in lingua originale. Si consiglia di fare riferimento al Glossario Cloud Native della CNCF, al Platforms WG Glossary di questo gruppo di lavoro e alla restante letteratura in materia per ogni dubbio e approfondimento.