Paolo Dalprato

Configurazione di Claude

Ultimo aggiornamento: 13 maggio 2026

Questo manuale descrive la configurazione di Claude, articolata su più livelli, e le regole che la tengono insieme. Il manuale tratta le interfacce consumer, claude.ai, Claude Desktop e l'app per smartphone, l'uso delle API resta fuori.

```
La configurazione è il modo in cui l'utente personalizza il proprio rapporto con Claude. Non è un passo obbligatorio, ma fatta correttamente migliora la qualità degli scambi e di quello che Claude produce. Le regole stabiliscono ad esempio lingua e tono, gli strumenti che Claude attiva, i momenti in cui chiede conferma e quelli in cui procede in autonomia. Possono definire anche le assunzioni che Claude fa sull'utente e su cosa sta cercando di ottenere.
```

Il problema è che queste regole non si scrivono in un unico punto. Oltre a chiedersi *come scrivere una regola*, occorre chiedersi *dove scriverla*. E occorre anche chiedersi se ha senso usare un certo livello, dato il modo in cui si lavora con Claude.

Occorre conoscere la struttura dei vari livelli che compongono l'architettura della configurazione, oltre ad adottare un'impostazione coerente per la scrittura delle regole.

Come è organizzato

Il manuale procede in tre tempi.

- **La prima parte presenta i livelli**. Il capitolo 1 descrive i quattro livelli di configurazione e introduce la portabilità, un tema che torna in tutti i capitoli successivi. Il capitolo 2 affronta le regole di prevalenza fra livelli, cioè come si decide quale regola prevale quando due istruzioni dicono cose diverse.
- **La seconda parte tratta i quattro livelli uno per uno**. Le istruzioni generali, le istruzioni di progetto, le istruzioni di Cowork e gli stili di scrittura. Per ciascun livello il manuale spiega quando si usa, cosa ha senso scriverci, propone un template di configurazione commentato, e raccoglie gli errori ricorrenti. La parte chiude con il capitolo 7, che mostra come i livelli collaborano nel loro insieme, descrive i profili tipici di configurazione a seconda di come si usa Claude, e tratta la manutenzione della configurazione nel tempo.
- **La terza parte va oltre i singoli livelli**. Il capitolo 8 mappa le altre fonti che concorrono al comportamento di Claude, dalle skill ai server MCP, dalle iniezioni di sistema alla memoria, e che restano fuori dai quattro livelli configurabili. Il capitolo 9 tratta la forza delle regole, cioè il modo in cui la

formulazione e il contesto rendono una regola più o meno efficace, oltre alla sua collocazione in un certo livello.

A chi è rivolto

Il manuale si rivolge a chi usa Claude per lavoro e vuole ottenere risposte coerenti nel tempo, in ambienti e progetti diversi. Non richiede competenze tecniche specifiche. Richiede di accettare che la configurazione di un assistente non è qualcosa che si fa una volta sola. È un'attività che si aggiusta nel tempo, a partire da osservazioni concrete sul proprio modo di lavorare.

Un'impostazione di lettura

I capitoli sono pensati per essere letti in sequenza, ma ciascuno è abbastanza autonomo da poter essere consultato da solo. Il capitolo 1 conviene leggerlo per primo in ogni caso, perché stabilisce il vocabolario e i confini del tema. I capitoli sui singoli livelli possono essere consultati all'occorrenza, quando si vuole rivedere una configurazione esistente. I capitoli 7, 8 e 9 chiudono il quadro. Sono utili per chi ha già operato sui quattro livelli e vuole capire perché alcune configurazioni si comportano in modo inatteso rispetto alle sole regole di prevalenza.

Indice

La struttura dei livelli di configurazione

- I quattro livelli
- La portabilità tra ambienti
- Note pratiche sulla scrittura

Le regole di prevalenza

- Stratificazione, non sostituzione
- Conflitto e combinazione
- La trasversalità degli stili
- L'istruzione di chat come ultima parola
- Una nota sulla fonte
- Le due dimensioni della prevalenza

Le istruzioni generali

- Quando vengono lette
- Criteri di collocazione
- Esempi di collocazione
- Un template di configurazione commentato
- Anti-pattern ricorrenti

Le istruzioni di progetto

- Quando vengono lette
- La stessa istruzione, ambienti diversi
- Le regole di prevalenza alla prova del contesto
- Criteri di collocazione
- Esempi di collocazione
- Un template di configurazione commentato

- Anti-pattern ricorrenti

Le istruzioni Cowork

- Quando vengono lette
- Criteri di collocazione
- Esempi di collocazione
- Un template di configurazione commentato
- Anti-pattern ricorrenti

Gli stili di scrittura

- Quando si applicano
- Criteri di collocazione
- Esempi di collocazione
- Stili predefiniti e stili personalizzati
- Come si combinano con gli altri livelli
- Un template di stile commentato
- Anti-pattern ricorrenti

La collaborazione fra i livelli

- Come si combinano le configurazioni
- Configurazioni globali per profilo
- Promozione e retrocessione delle regole
- Convivenza e ridondanza
- Scenari di configurazione completa

Altre fonti di configurazione

- Le skill come fonte di configurazione
- I server MCP e le loro descrizioni
- Le iniezioni di sistema
- Memoria e chat passate
- Come riconoscere e gestire queste fonti

La forza delle regole

- I fattori interni alla formulazione
- Quando la forza ribalta la gerarchia
- Scrivere regole forti dove serve
- Un caso operativo verificato

La struttura dei livelli di configurazione

La configurazione di Claude si articola su quattro livelli. Sono i punti in cui l'utente può intervenire per personalizzare il comportamento dell'assistente, dalla lingua al tono agli strumenti che attiva. I livelli sono distinti per ambito e regole di funzionamento. Tre formano una gerarchia, dal più generale al più specifico. Il quarto è un livello a sé, non interviene sul comportamento di fondo ma sulla forma delle risposte.

Le differenze fra i livelli non riguardano solo dove si scrivono le regole, ma anche quanto sono ampie le situazioni in cui valgono e come si combinano fra loro. Mettere ogni cosa nello stesso campo riduce l'efficacia di tutte le regole, e in qualche caso le annulla. Da qui la scelta su dove collocare ciascuna regola.

```
Una nota di contesto. Il manuale tratta i livelli disponibili nelle interfacce consumer
di Claude, cioè claude.ai, Claude Desktop e l'app per smartphone. L'uso delle API resta
fuori.
```

Questo capitolo presenta i quattro livelli uno per uno, con l'ambito di ciascuno e una prima nota sulle ragioni della separazione. Tratta poi la portabilità, un tema che torna in più punti del manuale. Chiude con due indicazioni pratiche, una sulla scrittura delle istruzioni e una sull'uso di Claude come aiuto nella stesura iniziale.

I quattro livelli

I quattro livelli sono le istruzioni generali, le istruzioni di progetto, le istruzioni di Cowork e gli stili di scrittura. I primi tre formano una gerarchia, valgono per ambiti progressivamente più ristretti e possono contraddirsi fra loro. Il quarto è un livello trasversale, sta fuori dalla gerarchia ma incide ugualmente sul comportamento dell'assistente.

Le **istruzioni generali** sono il livello base della configurazione. Si trovano in *Impostazioni*, sezione *Profilo*, sia su claude.ai sia in Claude Desktop, e si applicano anche alle app per smartphone. Sono un campo di testo libero salvato lato server e legato all'account, quindi valgono in ogni nuova conversazione, indipendentemente dall'ambiente e dal dispositivo da cui viene avviata. È il posto in cui scrivere le regole che devono valere sempre, qualunque sia il contesto di lavoro.

Le **istruzioni di progetto** sono una configurazione specifica del singolo progetto. I progetti esistono in quattro ambienti, claude.ai, sia web che mobile, l'area Chat di Claude Desktop e l'area Cowork dello stesso Desktop. Claude.ai e l'area Chat condividono lo stesso spazio server, quindi i progetti creati in un ambiente compaiono anche nell'altro e le istruzioni associate sono le stesse. Va tenuto presente che le interfacce non coincidono al cento per cento nelle funzionalità disponibili. La stessa istruzione di progetto può produrre esiti diversi a seconda di dove si apre la conversazione.

I progetti Cowork, anche se vivono dentro Claude Desktop, sono un gruppo separato. Hanno caratteristiche in più trattate nel capitolo 5, in particolare la possibilità di programmare attività ricorrenti e la gestione di agenti che operano in modo autonomo. Gli agenti possono anche essere strutturati in catene a più livelli, quando un agente a sua volta delega a sub-agenti. In tutti i casi le istruzioni di progetto si aggiungono a quelle generali e le specializzano per quel contesto. Possono aggiungere nuove regole, affinare le esistenti o prevalere su di esse quando il progetto richiede un comportamento diverso. Si pensi a un progetto relativo a una committenza internazionale in cui tutti i documenti devono essere in inglese, mentre nelle istruzioni generali la lingua richiesta è l'italiano.

Le **istruzioni di Cowork** sono un livello che esiste solo nell'area Cowork di Claude Desktop. Si configurano dal menu *Impostazioni*, nella sezione *Cowork*, accessibile solo da Claude Desktop. Si applicano a tutte le sessioni Cowork e si sommano alle istruzioni generali senza sostituirle. Raccolgono le regole utili nel lavoro su Cowork, che altrove sarebbero fuori posto. Comprendono in particolare le regole sulla gestione degli agenti, perché Cowork è il solo ambiente consumer in cui Claude può delegare compiti ad agenti autonomi. Le regole più specifiche sull'uso degli agenti possono comparire anche nelle istruzioni dei singoli progetti Cowork, quando riguardano un ambito di lavoro circoscritto.

Gli **stili di scrittura** sono il livello trasversale. Non istruiscono Claude su come comportarsi in generale, ma sullo stile delle risposte, intervenendo su formato, tono e registro. Si attivano per conversazione, si disattivano, si cambiano. Non competono con gli altri livelli, si sovrappongono a essi. Esistono perché capita di voler cambiare il modo di scrivere di Claude senza toccare le regole di comportamento sottostanti.

I meccanismi che regolano la coesistenza fra livelli, e in particolare le regole di prevalenza in caso di conflitto, sono trattati nel capitolo 2. Da qui in avanti il manuale li richiama quando necessario, dando per acquisito il quadro che il prossimo capitolo descrive in modo unitario.

La portabilità tra ambienti

Buona parte di queste configurazioni è legata all'account, non al dispositivo. È una caratteristica con un risvolto pratico che vale la pena anticipare. Le istruzioni generali seguono l'utente ovunque, si ritrovano identiche su claude.ai da qualsiasi computer, in Claude Desktop su più macchine, nelle app per smartphone. Lo stesso vale per i progetti di claude.ai e dell'area Chat di Claude Desktop e per le loro istruzioni. È una comodità, ma il loro uso va valutato con attenzione.

La prima cosa da considerare riguarda i riferimenti a file locali. Le app per smartphone e claude.ai non hanno accesso al filesystem, quindi qualsiasi istruzione che punti a un percorso del tipo `D:\Documenti\…` funziona solo in Claude Desktop, e solo sul computer in cui quel percorso esiste realmente. Quando si lavora su più macchine con Claude Desktop installato, la struttura di cartelle deve essere coerente fra loro. Altrimenti la stessa istruzione si comporta bene su un pc e fallisce sull'altro.

Il problema si estende ai server MCP locali, utilizzabili solo in Claude Desktop e quindi invisibili a claude.ai e alle app per smartphone. Anche qui, se Claude Desktop viene usato su più pc, occorre verificare che i server MCP siano installati e configurati in modo uniforme.

Il discorso per le skill è simile. Anche le skill si dividono in due gruppi, quelle locali che vivono sul filesystem di un singolo pc e quelle legate all'account, che seguono l'utente ovunque. Una skill può essere di account solo se è effettivamente portabile. Non deve richiamare percorsi locali, server MCP non disponibili in tutti gli ambienti, o altri elementi specifici di una singola installazione. Quando queste condizioni non sono rispettate la skill va tenuta locale, e ricade nella stessa logica dei file e dei server MCP discussa sopra.

Ogni volta che una configurazione di account richiama qualcosa che esiste solo localmente, come file, server MCP o skill, conviene chiedersi se debba davvero valere ovunque. Se la risposta è no, la regola va spostata su un ambito più circoscritto, come le istruzioni di un progetto usato solo in Claude Desktop.

Note pratiche sulla scrittura

Le istruzioni sono campi di testo libero, ma conviene scriverle in formato markdown. L'interfaccia di Claude riconosce la sintassi, gli H2 diventano sezioni leggibili, le liste sono interpretate come tali, i blocchi di codice sono segnalati come tali. Una configurazione strutturata con titoli di sezione (`## Profilo`, `## Metodo`, e così via) si legge più facilmente. Vale sia per l'utente quando rilegge e aggiorna il testo, sia per Claude quando compone il contesto della conversazione. Niente vieta di scrivere un testo piatto, ma la struttura aiuta, e non costa nulla.

Scrivere da zero una configurazione efficace, specie le prime volte, può essere faticoso. Per fortuna Claude stesso può essere un buon aiuto in questa fase, anche già nella fase di analisi. Basta descrivere l'ambito di lavoro e le preferenze in forma conversazionale, e chiedere una prima bozza di istruzioni. Oppure chiedere di rivedere una bozza esistente, segnalando ridondanze, ambiguità e regole che sarebbero meglio collocate su un altro livello. La prima bozza non è mai quella definitiva, ma è un punto di partenza migliore del foglio bianco.

Sul versante opposto, quando una configurazione è già ampia e si vuole verificarla, esiste uno strumento dedicato che ho sviluppato. Si chiama Claude Config Audit (https://www.ai-know.pro/claude-config-audit/), è una skill open source che analizza server MCP, plugin, skill e istruzioni. Cerca duplicazioni, conflitti, riferimenti a strumenti non più presenti e altre incoerenze tipiche delle configurazioni cresciute nel tempo. Non è indispensabile per iniziare, è utile quando il setup ha qualche mese di vita alle spalle e vale la pena fare manutenzione.

Il riferimento ufficiale Anthropic alle funzionalità di personalizzazione, in inglese, è disponibile alla pagina di supporto dedicata (https://support.claude.com/en/articles/10185728-understanding-claude-s-personalization-features).

Le regole di prevalenza

I livelli di configurazione descritti nel capitolo precedente non operano da soli. Quando si apre una conversazione, Claude legge insieme tutti i livelli pertinenti e ne ricava il prompt di sistema effettivo. Le istruzioni dei diversi livelli si compongono nel prompt come testo concatenato, e da quel momento agiscono insieme per la durata della conversazione. La domanda è cosa succede quando due livelli dicono cose contraddittorie, e come si risolve la prevalenza. Questo capitolo riguarda le regole che governano la composizione fra livelli, mentre le specificità di ciascuno verranno descritte nei capitoli successivi.

```
Escludendo gli stili di scrittura le modifiche apportate a un qualunque livello diventano attive nelle conversazioni avviate dopo il salvataggio. Le conversazioni già aperte continuano con la versione letta al loro avvio, perché è in quel momento che il prompt di sistema viene composto. Per applicare subito una nuova regola a una conversazione in corso va richiamata esplicitamente nella richiesta successiva, oppure va avviata una conversazione nuova.
```

Le istruzioni non sono comandi che Claude esegue meccanicamente, sono contesto che entra nel prompt di sistema della conversazione insieme ad altri elementi, Claude interpreta l'insieme per produrre le risposte. La prevalenza di una regola su un'altra non è il risultato di una catena di se-allora, dipende da come il modello integra tutti i segnali che riceve. Le regole di prevalenza che il capitolo descrive sono il modo in cui questa integrazione si comporta nei casi normali, cioè quando tutte le istruzioni hanno una formulazione di pari intensità.

```
Tutto il capitolo lavora a parità di forza fra le istruzioni. Quando le istruzioni in conflitto sono espresse con intensità decisamente diverse, può accadere che la regola di prevalenza per posizione gerarchica si ribalti. È una dimensione ulteriore della composizione, trattata in modo completo al capitolo 9. Fino a quel punto la regola «a parità di forza il livello più specifico prevale» orienta correttamente le scelte di configurazione, perché copre la maggior parte dei casi pratici.
```

Stratificazione, non sostituzione

Le regole dei diversi livelli non si sovrascrivono come si sovrascrivono i file. La metafora corretta è quella della stratificazione. Ogni livello più specifico si appoggia su quelli precedenti. Aggiunge le proprie regole, le affina dove serve, e in caso di contraddizione prevale nell'ambito che lo riguarda. I livelli sottostanti però non spariscono. Continuano a essere attivi e a fare da fondamento.

Un esempio. Le istruzioni generali chiedono di rispondere in italiano. Si crea un progetto per una committenza internazionale che richiede l'inglese, e nelle sue istruzioni si scrive «rispondi in inglese». Nelle chat di quel progetto Claude scrive in inglese. Fuori del progetto Claude continua a scrivere in italiano, perché la regola generale non è stata modificata, è stata solo sovrastata localmente. Eliminare la regola

generale per evitare la contraddizione è inutile, e finisce per peggiorare le cose. Lascia priva di indicazione tutta la parte di lavoro che si svolge fuori del progetto.

La stessa logica vale per i livelli intermedi. Le istruzioni di progetto si appoggiano sulle generali. Le istruzioni di Cowork si appoggiano sulle generali quando si lavora in Cowork, e si compongono con le istruzioni di progetto Cowork quando si è dentro un progetto. Gli stili di scrittura, trattati nel capitolo 6, hanno una posizione un po' diversa che si vedrà più avanti in questo capitolo.

Conflitto e combinazione

La regola di prevalenza è semplice nell'enunciato. Quando due livelli si contraddicono direttamente sullo stesso aspetto, prevale il livello più specifico. La regola di un progetto vince sulla regola corrispondente delle istruzioni generali, perché il progetto è stato creato apposta per un contesto che richiede un comportamento diverso. Lo stesso vale fra Cowork e generali, fra istruzioni di progetto Cowork e istruzioni di Cowork.

Riprendendo l'esempio della lingua, in quel progetto Claude scrive in inglese. Le istruzioni di progetto prevalgono anche per il registro, per la lunghezza, per l'attivazione di una skill. Quando nelle generali e in un livello più specifico si dice qualcosa di diverso sullo stesso punto, la versione del livello più specifico è quella che prevale.

I casi di conflitto vero sono però la minoranza. La situazione più comune è che le regole dei diversi livelli parlano di cose diverse. In quel caso non c'è bisogno di scegliere fra loro. Si sommano. Le istruzioni generali chiedono italiano, le istruzioni di un progetto chiedono registro formale. Le due regole non si contraddicono, una riguarda la lingua, l'altra il registro. In quel progetto Claude risponde in italiano formale, sommando le indicazioni dei due livelli senza che nessuna debba cedere all'altra.

La distinzione fra conflitto e combinazione non è sempre evidente alla prima lettura. Aiuta chiedersi se le due regole rispondono allo stesso tipo di domanda. «In che lingua scrivere?» è una domanda. «Con che registro?» è un'altra. Regole che rispondono alla stessa domanda confliggono, e la prevalenza fra livelli decide quale vince. Regole che rispondono a domande diverse si combinano semplicemente.

La trasversalità degli stili

Gli stili di scrittura non si inseriscono nella catena gerarchica, hanno una natura diversa. Sono un livello trasversale che si attiva sopra qualsiasi configurazione, indipendentemente dal progetto in cui si lavora. L'utente può cambiarli in qualsiasi momento, anche a metà conversazione. Per questo si parla di livello trasversale alla gerarchia, non di livello superiore o inferiore agli altri. In Desktop agiscono anche in Cowork ma con modalità diverse da Chat, questa parte verrà approfondita nel capitolo specifico sugli stili.

In caso di conflitto fra uno stile attivo e una regola di un livello sottostante, lo stile prevale. È il livello più granulare fra quelli configurati persistentemente. Agisce per singola conversazione e può essere attivato, disattivato o cambiato dall'utente in qualsiasi momento. La sua granularità più fine lo rende più specifico, e

la regola «il più specifico prevale» continua a valere. La specificità però è di un altro tipo, granulare invece che gerarchica.

Un esempio. Le istruzioni generali chiedono di evitare il grassetto e usare il corsivo per l'enfasi. Si attiva uno stile che prevede l'uso del grassetto. Le risposte mostreranno il grassetto finché lo stile è attivo. Quando si torna al default, l'indicazione delle generali torna ad applicarsi. La regola generale non è stata modificata, è stata sovrastata localmente, esattamente come succede fra livelli gerarchici.

Nella maggior parte dei casi però gli stili non confliggono con i livelli sottostanti. Intervengono su aspetti della formulazione che si combinano con le regole già presenti. Lo stile *Formale* aggiunge un registro, non rimuove la lingua decisa nelle generali. La situazione più frequente è quella della combinazione, non quella del conflitto, esattamente come per i livelli gerarchici.

L'istruzione di chat come ultima parola

Le istruzioni date nel corso di una conversazione si collocano al livello più specifico che esista, quello del singolo turno. Non sono configurazioni persistenti, non sopravvivono alla chat in cui sono state scritte. All'interno di quella chat però hanno la priorità più alta. Battono qualsiasi configurazione precedente, le istruzioni generali, quelle di progetto, quelle di Cowork e gli stili.

L'esempio è semplice. Un progetto richiede l'inglese e Claude sta rispondendo in inglese. In una chat di quel progetto l'utente scrive «per questa risposta scrivi in italiano». Claude risponde in italiano. La regola di progetto non viene modificata, continua a valere per le risposte successive. Solo quel turno, o il tratto di conversazione esplicitamente coperto dall'istruzione, segue l'indicazione data direttamente in chat.

La logica è la stessa già vista. Il livello più specifico prevale, e l'istruzione di chat è il livello più specifico in assoluto. Non c'è nulla di più granulare, e nessuna configurazione persistente può batterla, perché la sua specificità è temporale prima ancora che strutturale. Vale qui e adesso.

Una nota sulla fonte

La pagina ufficiale di supporto Anthropic descrive cosa fa ciascun livello di personalizzazione, ma non pubblica un ordine di prevalenza esplicito per i livelli trattati nel manuale. La sola prevalenza dichiarata esplicitamente da Anthropic riguarda le istruzioni di organizzazione, disponibili nei piani Team ed Enterprise e fuori dall'ambito consumer del manuale. Le regole descritte fin qui sono coerenti con il modello di funzionamento del prompt di sistema concatenato e con l'esperienza condivisa dalla community. Sono coerenti anche con un'osservazione che Anthropic stessa fa nella documentazione, quando dichiara che la prioritizzazione delle istruzioni è una proprietà del prompt e non un meccanismo deterministico. Anthropic raccomanda di verificare gli effetti delle proprie istruzioni nelle conversazioni reali, è il suggerimento che il manuale fa proprio.

Le due dimensioni della prevalenza

Quanto detto fin qui descrive una sola dimensione della prevalenza, quella che dipende dalla collocazione gerarchica delle istruzioni. Il livello più specifico vince, perché è stato scritto apposta per un contesto che richiede un trattamento particolare. È la dimensione che il manuale userà come criterio guida nei capitoli successivi sui singoli livelli, e che orienta correttamente la maggior parte delle scelte di configurazione.

Come anticipato all'inizio del capitolo esiste però una seconda dimensione, che si manifesta quando le istruzioni in conflitto sono formulate con intensità diverse. Un'istruzione netta può prevalere su una preferenza morbida, anche se la prima si trova in un livello inferiore. Una regola ribadita in più punti del prompt può avere più forza di una regola scritta una volta sola in una posizione formalmente più specifica. La forza di un'istruzione, intesa come intensità della formulazione e rinforzo dal contesto, è una variabile autonoma rispetto alla collocazione gerarchica.

Le due dimensioni convergono quando la regola scritta nel livello più specifico è anche quella formulata con maggiore intensità. In questo caso il vincitore è chiaro, e non ci sono sorprese. Possono però divergere, se l'istruzione più forte sta in un livello meno specifico la prevalenza può ribaltarsi rispetto a quanto la sola regola gerarchica farebbe prevedere. Il quadro completo di questa seconda dimensione, con i fattori che concorrono a determinare la forza e un caso operativo verificato, è al capitolo 9.

Da qui in avanti, fino a quel capitolo, il manuale lavora a parità di forza fra le istruzioni in conflitto. È un'assunzione che semplifica l'esposizione e che corrisponde alla situazione reale nella maggior parte delle configurazioni scritte con disciplina. Quando si presenteranno casi in cui la forza diventa rilevante, i capitoli sui singoli livelli ne daranno indicazione, lasciando l'esposizione completa al capitolo conclusivo.

```
## In sintesi

Cinque principi governano la composizione fra livelli, a parità di forza delle
istruzioni.

1. I livelli si stratificano, non si sovrascrivono. Una regola di un livello specifico
non cancella la regola corrispondente di un livello più generale, la sovrasta solo
nell'ambito che la riguarda.

2. Quando due livelli dicono cose diverse sullo stesso aspetto è conflitto, e prevale il
livello più specifico. Quando parlano di aspetti diversi non c'è conflitto, le regole si
sommano.

3. Gli stili stanno fuori dalla gerarchia ma seguono la stessa logica del più specifico,
con specificità di tipo granulare invece che gerarchico.

4. L'istruzione data direttamente in chat è il livello più specifico in assoluto.

5. Quando le istruzioni hanno intensità molto diverse entra in gioco una seconda
dimensione, la forza delle istruzioni, trattata al capitolo 9.
```

Le istruzioni generali

Le istruzioni generali sono il livello base della configurazione, presentato al capitolo 1 come il fondamento su cui si appoggiano gli altri livelli. Si configurano nell'area *Impostazioni*, sezione *Profilo*, in un campo introdotto dall'indicazione «*Quali preferenze personali dovrebbe considerare Claude nelle risposte? Le tue preferenze si applicheranno a tutte le conversazioni, nel rispetto delle linee guida di Anthropic*». Il campo è salvato lato server e legato all'account, e vale in ogni nuova conversazione indipendentemente dall'ambiente e dal dispositivo.

```
Scriverle bene ha un effetto a cascata, perché semplifica la configurazione dei livelli
più specifici, che possono dare per assodato il default e limitarsi a esprimere le
differenze.
```

Il capitolo descrive il funzionamento delle istruzioni generali, con quale criterio inserire una regola in questo livello e cosa lasciare altrove. Propone un template come punto di partenza e raccoglie gli anti-pattern ricorrenti.

Quando vengono lette

Le istruzioni generali vengono lette all'avvio di ogni nuova conversazione, da claude.ai, da Claude Desktop o dall'app per smartphone.

Le istruzioni sono contesto e non comandi, come visto al capitolo 2. Una conseguenza pratica è che istruzioni contraddittorie al loro interno, ambigue o troppo verbose ne rendono complessa l'applicazione. Una buona configurazione di questo livello è compatta, priva di ripetizioni interne e priva di regole che si annullano a vicenda. È un'osservazione che vale per tutti i livelli, ma per le generali conta di più, perché qui ogni regola entrerà in ogni conversazione futura.

Criteri di collocazione

Il criterio guida è che qui vanno le regole che devono essere applicate in ogni conversazione, a prescindere dal progetto, dall'ambiente e dallo strumento in uso in quel momento. Tutto ciò che è specifico di un ambito appartiene a livelli più circoscritti.

Rientra bene nelle istruzioni generali il profilo personale dell'utente, nella misura in cui influenza il modo in cui Claude risponde, con il ruolo professionale, l'ambito di lavoro abituale, i riferimenti tematici ricorrenti. Rientrano le preferenze linguistiche e tonali di default, tipicamente la lingua desiderata, il registro preferito, il livello di brevità o di approfondimento. Rientrano le metodologie di default, per esempio il modo in cui Claude deve verificare i fatti, segnalare le assunzioni non verificate, distinguere fra affermazioni verificate e

interpretazioni plausibili. Rientrano infine le istruzioni di base sull'uso degli strumenti e delle skill, nei limiti in cui valgono per tutti i lavori dell'utente.

Una trappola frequente, specie per chi ha più ambiti di lavoro diversi, è l'accumulo a questo livello di regole provenienti da tutti questi ambiti. Si confida che Claude capisca quando applicare cosa. Il risultato è una configurazione che si contraddice al suo interno e che diluisce l'efficacia di tutte le regole, incluse quelle scritte bene. Conviene tenere le istruzioni generali come un profilo di base leggero e coerente, e spostare le specificità nei progetti.

Esempi di collocazione

Tre esempi rendono concreto il criterio della pervasività.

- Il primo riguarda il metodo di lavoro. Le istruzioni generali richiedono a Claude di verificare le fonti, segnalare le assunzioni non verificate, distinguere fra affermazioni verificate e interpretazioni plausibili. È una regola di metodo che vale per la maggior parte dei lavori dell'utente, e ha quindi senso al livello più generale. In un progetto esplicitamente dedicato al brainstorming creativo, dove il rigore metodologico ostacola più di quanto aiuti, la regola può essere allentata da un'istruzione di progetto, senza intaccare il default. Negli altri progetti il metodo torna ad applicarsi, perché la regola generale non è stata modificata, è stata solo sovrastata localmente nel progetto di brainstorming.
- Il secondo riguarda il profilo dell'utente. Il ruolo professionale, l'ambito di lavoro abituale, i temi ricorrenti che caratterizzano la propria attività sono informazioni di profilo che orientano Claude in qualsiasi conversazione, e stanno bene nelle generali. Quando si lavora per un cliente specifico, le caratteristiche di quel cliente, il glossario interno, le persone con cui ci si interfaccia, sono informazioni di contesto specifiche di quel rapporto e vanno nelle istruzioni di un progetto dedicato al cliente. La distinzione è fra ciò che identifica il professionista in modo stabile e ciò che caratterizza una committenza fra le tante.
- Il terzo riguarda una preferenza che sembra di default ma non lo è. Un esempio è la lunghezza preferita delle risposte. Per la maggior parte del lavoro va bene una lunghezza media, e la regola può stare nelle generali. Esistono però lavori che richiedono risposte sistematicamente più estese, ad esempio i contesti editoriali in cui si producono articoli completi. La scelta è fra due strade. Si scrive la lunghezza media nelle generali e la si specializza nel progetto editoriale con la lunghezza più estesa. Oppure non si scrive nulla nelle generali e si tiene la regola solo nei progetti che ne hanno bisogno. La direzione dipende da quante volte la lunghezza media è effettivamente preferita. Se è il caso predominante, scriverla nelle generali è coerente con la pervasività. Se è una scelta che vale per pochi progetti, scriverla solo nei progetti che ne hanno bisogno tiene le generali più pulite.

I tre esempi seguono lo stesso schema. Nelle generali va ciò che vale per la maggior parte del lavoro. Le specificità di un singolo contesto stanno nei livelli più circoscritti. Le eccezioni rispetto al default si scrivono come variazioni nel livello specifico, lasciando il livello base intatto.

Un template di configurazione commentato

Il template che segue è pensato come proposta di partenza, non come modello da replicare. Il criterio di fondo è quello di cui si è parlato, la pervasività. Le sezioni sono indicative, possono essere fuse, divise o rinominate in base alle proprie abitudini. Quello che conta è che ogni blocco contenga solo regole che valgono davvero in ogni conversazione.

```
## Profilo

[Chi è l'utente dal punto di vista professionale, il ruolo, l'ambito di lavoro, i temi
principali su cui opera. Solo gli elementi pervasivi, non le specificità dei singoli
progetti.]

## Preferenze di default

[Lingua, tono, registro, livello di brevità o di approfondimento
desiderati come default. Queste preferenze possono essere
specializzate o sovrascritte da livelli più specifici.]

## Metodo

[Come Claude deve gestire le fonti, le assunzioni non verificate,
la distinzione fra fatti, interpretazioni plausibili e speculazioni.
Regole metodologiche trasversali.]

## Strumenti e skill

[Indicazioni sull'uso di strumenti e skill che valgono per tutti
i lavori dell'utente. Nulla che presupponga un ambiente o un
dispositivo specifico.]
```

Come si vede le quattro sezioni coprono le famiglie di regole che ha senso tenere a questo livello e lasciano fuori tutto il resto.

La sezione *Profilo* non è un curriculum. È il minimo che serve a Claude per capire il contesto del suo interlocutore, quindi poche righe orientative, non un elenco esaustivo di competenze.

La sezione *Preferenze di default* raccoglie le scelte di forma, con l'accortezza di non scrivere regole che si sa già di voler modificare spesso nei progetti. In quei casi è meglio mettere la regola direttamente nelle istruzioni di progetto.

La sezione *Metodo* è quella che porta più valore, ed è anche la più trascurata dagli utenti alle prime armi, perché è la più astratta. È il posto in cui si descrive non cosa Claude deve fare, ma come deve affrontare il lavoro.

La sezione *Strumenti e skill* va tenuta leggera. Qualsiasi regola che dipenda dall'ambiente appartiene a un altro livello.

Anti-pattern ricorrenti

Alcuni errori si ripetono con sufficiente frequenza da meritare una menzione esplicita.

- **L'accumulo di regole provenienti da ambiti diversi**. Quando si lavora su più progetti, ciascuno con le sue specificità, nel dubbio si tende a inserire tutto nelle istruzioni generali, confidando che Claude capisca di volta in volta quale regola applicare. Il risultato è una configurazione che si contraddice al suo interno e che perde efficacia anche sulle regole scritte bene, perché il contesto diventa rumoroso e le priorità relative si sfocano. La soluzione è spostare le specificità nei progetti, lasciando qui solo ciò che vale sempre.
- **La dipendenza implicita dall'ambiente**. Vengono inserite istruzioni che presuppongono un percorso del filesystem, un server MCP, una skill locale, senza rendersi conto che l'istruzione arriverà identica su claude.ai, su Desktop locale e sullo smartphone. In due ambienti su tre non funzionerà. La soluzione non è rinunciare a quelle istruzioni, è collocarle nel livello giusto. Il posto naturale sono le istruzioni di un progetto dedicato al lavoro su filesystem, sapendo che valgono solo in Claude Desktop e sul computer che ha quella struttura.
- **L'eccesso di lunghezza e ridondanza**. Le istruzioni generali non sono un documento contrattuale, sono un contesto che viene letto e interpretato. Un testo molto lungo con regole ripetute, sinonimi che dicono cose leggermente diverse, eccezioni dentro eccezioni, produce un'interpretazione meno accurata di un testo più breve e pulito. La revisione periodica di questa configurazione, per rimuovere ciò che non serve più o riformulare ciò che si è chiarito nel tempo, è parte dell'igiene di lavoro.

Le istruzioni di progetto

Le istruzioni di progetto sono il livello di configurazione che specializza il comportamento di Claude all'interno di un singolo progetto. Il concetto di progetto si trova in claude.ai, sia web che mobile, e in Claude Desktop, sia nell'area Chat che nell'area Cowork. In ciascuno di essi un progetto dispone di un campo dedicato alle istruzioni di progetto, distinto dal prompt delle singole chat. A differenza del campo *Profilo* delle istruzioni generali, qui non c'è una descrizione persistente che resti visibile a campo compilato. Il campo mostra solo un placeholder che scompare al primo inserimento e che varia leggermente da ambiente a ambiente, pur mantenendo lo stesso significato di fondo. Nei progetti Cowork compare ad esempio la formula «*Aggiungi tono, formattazione o regole per guidare il funzionamento di Claude*».

```
Questo capitolo si concentra sul testo delle istruzioni di progetto. Non tratta i file
caricati nel progetto, che sono materiale di lavoro a disposizione di Claude ma non sono
istruzioni esplicite. Non tratta nemmeno la memoria di progetto, un meccanismo separato
di accumulo automatico di informazioni a partire dalle conversazioni. Sono tutti elementi
che definiscono l'esperienza dei progetti, ma operano su piani diversi con scopi diversi.
```

Il capitolo indica cosa ha senso scrivere a questo livello e cosa lasciare altrove. Propone un template come proposta di partenza e raccoglie gli anti-pattern ricorrenti.

Quando vengono lette

Le istruzioni di progetto vengono lette all'avvio di ogni nuova chat aperta dentro il progetto, e si compongono nel prompt di sistema della conversazione insieme alle istruzioni generali. Le chat avviate fuori dal progetto non leggono queste istruzioni, il loro prompt di sistema contiene solo il livello base.

Due aspetti tecnici cambiano il modo in cui conviene configurarle.

- **Gli ambienti in cui i progetti vivono**. Claude.ai e l'area Chat di Claude Desktop condividono lo stesso spazio server, quindi un progetto creato in un ambiente compare anche nell'altro con le stesse istruzioni associate. I progetti Cowork sono invece un gruppo separato e locale e vivono solo dentro Claude Desktop. La separazione non è solo organizzativa, è strutturale, perché Cowork opera all'interno di un ambiente di esecuzione locale sul computer dell'utente. Il dettaglio è nel capitolo 5, qui basta tenere presente che un progetto Chat e un progetto Cowork sono cose diverse anche quando riguardano lo stesso lavoro.
- **Lo scarto fra spazio server condiviso e capacità effettive degli ambienti web e Desktop**. Lo stesso progetto Chat può essere aperto sia da claude.ai sia da Claude Desktop, e in entrambi i casi le istruzioni associate sono testualmente identiche. Quello che cambia è ciò che Claude riesce a fare con quel testo, perché le capacità a disposizione differiscono fra i due ambienti. La sezione successiva entra nel dettaglio di questo punto.

La stessa istruzione, ambienti diversi

Le istruzioni di progetto sono testo, e il testo è identico ovunque venga letto. Quello che cambia da un ambiente all'altro è ciò che Claude riesce effettivamente a fare con quel testo, perché le capacità a disposizione differiscono fra l'interfaccia web di claude.ai, l'area Chat di Claude Desktop e l'area Cowork dello stesso Desktop. La conseguenza è che la stessa istruzione di progetto, formalmente identica, può funzionare in modi diversi a seconda di dove la chat viene aperta. Capire questa differenza è particolarmente importante per i progetti che vivono nello spazio server condiviso fra web e Chat Desktop, dove la stessa istruzione viene effettivamente letta in entrambi gli ambienti.

Tre esempi chiariscono il punto.

- **Il primo riguarda una skill**. Le istruzioni di progetto contengono una regola del tipo «in questo progetto attiva sempre la skill di assistenza editoriale». Se la skill è registrata sull'account, l'istruzione funziona ovunque. Se invece è una skill locale, salvata nel filesystem di un singolo computer, funziona solo in Claude Desktop su quel pc. Sull'interfaccia web di claude.ai non funziona perché la skill non esiste in quell'ambiente. Lo stesso vale in Claude Desktop su un altro device in cui la skill non è installata. L'istruzione è scritta correttamente in entrambi i casi, ma l'effetto dipende dalla disponibilità della risorsa che richiama.
- **Il secondo riguarda un percorso del filesystem**. Le istruzioni di progetto contengono una regola del tipo «i materiali di questo progetto sono nella cartella `D:\Progetti\Cliente\`». Sull'interfaccia web di claude.ai l'istruzione è priva di effetto, perché il browser non ha accesso al filesystem del computer. In Claude Desktop l'istruzione è interpretabile, perché Desktop può leggere il filesystem locale tramite gli strumenti disponibili in Chat e in Cowork, con due avvertenze. Vale solo sul pc in cui quella struttura di cartelle esiste, su un altro computer la stessa istruzione si scontra con un percorso che non esiste. Inoltre l'accesso effettivo al filesystem dipende dai tool attivati e dalle loro configurazioni, che possono differire fra Chat e Cowork anche sullo stesso pc.
- **Il terzo è un controesempio utile**. Le istruzioni di progetto contengono una formulazione del tipo «rispondi sempre in modo conciso, evita preamboli, vai dritto al punto». Questa istruzione è agnostica rispetto all'ambiente di esecuzione, non fa riferimento a strumenti, file, server. L'effetto sarà lo stesso ovunque, web, Chat Desktop, Cowork, su qualsiasi pc. Le regole che riguardano lingua, tono, registro, livello di approfondimento, schema di risposta sono in genere portabili senza problemi, perché vivono interamente nel testo.

Da questi tre esempi si ricava un criterio operativo. Prima di scrivere una regola nelle istruzioni di progetto conviene chiedersi se la regola è autonoma rispetto all'ambiente, oppure se richiede risorse, percorsi o strumenti che vivono solo in alcuni ambienti. Le regole autonome stanno bene a questo livello, sono portabili e robuste. Le regole che dipendono da risorse locali stanno bene solo nei progetti destinati a un singolo ambiente, cioè Claude Desktop, e in quel caso vale la pena dichiararlo nella prima riga delle istruzioni, ad esempio scrivendo che «questo progetto è pensato per essere usato solo in Claude Desktop». Una nota di questo tipo non vincola Claude, vincola il lettore-autore quando rilegge la propria configurazione mesi dopo.

C'è anche un caso ulteriore, particolare di Cowork. Quando si crea un progetto Cowork si può scegliere di importare un progetto Chat come contesto. In quel caso le istruzioni del progetto Chat compaiono nelle impostazioni del progetto Cowork in modalità di sola lettura, e si affiancano al campo istruzioni proprio di Cowork che resta editabile. Il prompt di sistema effettivo della chat Cowork si compone così di tre blocchi, le istruzioni generali, le istruzioni del progetto Chat ereditato, le istruzioni del progetto Cowork. È una stratificazione interna e segue le stesse regole di prevalenza viste al capitolo 2. Per chi sa che un progetto Chat farà anche da base a un progetto Cowork, le istruzioni del progetto Chat vanno scritte tenendo conto che potranno arrivare in un ambiente con capacità più ampie, evitando però di anticipare regole che hanno senso solo in Cowork. Quelle stanno meglio nel campo istruzioni proprio di Cowork, lasciando il progetto Chat pulito sul proprio ambito. Il quadro completo è al capitolo 5.

Le regole di prevalenza alla prova del contesto

La regola «*a parità di forza il livello più specifico prevale*» è semplice nell'enunciato, e descrive abbastanza bene come si compongono i livelli di configurazione. Nella pratica vanno valutate con attenzione due dinamiche specifiche del livello di progetto che producono comportamenti che la sola regola gerarchica non basta a prevedere. La prima riguarda il rapporto fra come è formulata una regola di progetto e come è formulata la regola corrispondente nelle generali, nel caso la regola sia condivisa nei due livelli. La seconda riguarda il rapporto fra le istruzioni configurate e i segnali che arrivano dalla conversazione stessa.

Si pensi a una configurazione in cui le istruzioni generali chiedono di rispondere in italiano, accompagnando la richiesta con una motivazione lunga, ad esempio una nota che spiega il pubblico, l'ambito di lavoro, qualche caso d'uso. Nelle istruzioni di un progetto specifico c'è invece la regola secca «*rispondi in inglese*». La domanda è quale lingua verrà usata in quel progetto. La risposta coerente con quanto visto al capitolo 2 dice che prevale il livello più specifico, quindi l'inglese, perché la prevalenza è governata dalla specificità del livello e non dalla lunghezza del testo. La regola di progetto è scritta apposta per quel contesto, e questo basta.

Va detta però una sfumatura. L'asimmetria fra una motivazione articolata e un'istruzione minimale può introdurre un margine interpretativo, soprattutto in casi limite, ad esempio quando l'utente scrive in italiano e l'argomento è stato discusso in italiano nelle motivazioni delle generali. Non è un'inversione della prevalenza, ma si può manifestare una tendenza a propendere verso la lingua più presente nel contesto. Si previene scrivendo la regola di progetto in modo netto e univoco, e tenendo le motivazioni nelle generali brevi e funzionali. C'è anche un risvolto non tecnico, una motivazione lunga che convive con una regola secca trasmette all'autore una percezione di squilibrio quando rilegge la propria configurazione mesi dopo. La logica della stratificazione conviene affidarla alla struttura, non alla retorica.

La seconda dinamica è più interessante e tocca un punto che vale la pena rendere esplicito. Si immagini la stessa configurazione, italiano nelle generali, inglese nel progetto, e si supponga che nella chat del progetto l'utente scriva in italiano. Claude ha tre segnali simultanei, le generali chiedono italiano, il progetto chiede inglese, il messaggio appena ricevuto è in italiano. Il comportamento osservato in casi come questo è che Claude risponda in italiano, non in inglese. La regola di progetto, pur scritta correttamente e in linea con i meccanismi di prevalenza, perde nei fatti.

Non è un'eccezione alla regola, è il modo in cui Claude bilancia il prompt di sistema con il contesto della conversazione. La lingua usata dall'utente nel suo messaggio è essa stessa un'istruzione, implicita ma molto forte, e Claude tende a rispondere nella lingua dell'interlocutore. È lo stesso meccanismo che fa sì che, in assenza di istruzioni esplicite, l'assistente parli sempre la lingua del prompt che riceve. Quando l'istruzione di progetto richiede un comportamento contrario al segnale contestuale, le due forze si confrontano, e in molti casi vince il contesto.

Da qui due principi di metodo. Le istruzioni di progetto, come quelle generali, non sono comandi vincolanti, sono contesto che entra in dialogo con altro contesto. Quando una regola riguarda un elemento sensibile al contesto come la lingua, il tono, il livello di formalità, esprimerla una volta sola non basta a renderla rigida. Per ottenere il comportamento atteso anche di fronte a segnali contrari occorre formulare la regola in modo che anticipi esplicitamente il caso di deriva. Per la lingua, una formulazione del tipo «*rispondi sempre in inglese, anche se l'utente scrive in italiano*» resiste meglio dell'imperativo neutro «*rispondi in inglese*». La prima formulazione è chiusa rispetto al contesto, la seconda lascia spazio interpretativo alla conversazione.

Il secondo principio è di coerenza personale. Se nelle istruzioni di progetto si è scelta una lingua, conviene che anche i prompt nelle chat di quel progetto siano scritti in quella lingua. L'incoerenza fra istruzione configurata e comportamento dell'utente è essa stessa un segnale che indebolisce la regola. Una regola di progetto regge meglio quando l'utente agisce come se la regola fosse già operativa, senza chiedere a Claude di tenere ferma una direzione che chi scrive contraddice in pratica.

Quanto detto fin qui è stato anticipato nel capitolo 2 come dimensione della forza dell'istruzione e troverà esposizione completa al capitolo 9. Il livello di progetto è quello in cui la dimensione della forza si manifesta più spesso, perché è il primo livello in cui il conflitto fra istruzione configurata e contesto della conversazione diventa frequente.

Criteri di collocazione

Nelle istruzioni di progetto vanno solo le regole che riguardano quel progetto e che non avrebbero senso applicate ovunque. Tutto ciò che è generale appartiene alle istruzioni generali, tutto ciò che dura una conversazione appartiene alla chat o agli stili di scrittura.

Rientrano bene nelle istruzioni di progetto le specificità del contesto di lavoro che caratterizzano quel progetto. Il committente o il cliente per cui il progetto è stato creato, l'ambito tematico, gli interlocutori con cui si interagisce, il tipo di output atteso. Sono informazioni che orientano il modo in cui Claude legge le richieste e formula le risposte all'interno del progetto, e non hanno senso fuori da lì.

Rientrano bene anche le preferenze specifiche che si discostano dal default delle istruzioni generali. La lingua, se diversa da quella di default. Il registro, la lunghezza, il livello di approfondimento, il formato delle risposte. Tutto ciò che si vuole impostare diversamente rispetto al livello base, e solo per questo progetto. Quando si vuole una formattazione stabilmente diversa dal default per un intero progetto, il posto è qui. Gli stili di scrittura sono uno strumento più adatto alle variazioni che durano il tempo di una conversazione o di una serie breve di chat.

Rientrano bene le regole metodologiche specifiche. Se il progetto è di brainstorming, può avere senso allentare il rigore di verifica delle fonti che vale come default nelle generali. Se il progetto è di analisi rigorosa, può avere senso al contrario rinforzarlo. Se il progetto richiede una particolare attenzione a certi tipi di errore o a certe distinzioni concettuali, è corretto scriverlo qui.

Rientrano bene le indicazioni sugli strumenti e le skill che servono in quel progetto, purché siano disponibili negli ambienti in cui il progetto verrà aperto. Una skill di account che si vuole tenere attiva sempre nel progetto, un MCP da privilegiare per le ricerche, una libreria di riferimenti specifica. Vale la considerazione di portabilità della sezione *La stessa istruzione, ambienti diversi*. Le risorse locali funzionano solo nei progetti destinati a Claude Desktop, e in quel caso conviene segnalarlo nel testo.

Le istruzioni di progetto non vanno usati per compiti puntuali o richieste che vivono il tempo di una conversazione.

Una nota finale, la pulizia conta più della completezza, per cui una configurazione di progetto breve, asciutta, focalizzata su ciò che è davvero specifico, produce un comportamento più affidabile di una configurazione lunga che ripete cose già dette nelle generali. Il livello di progetto esiste per dire la differenza, non per ribadire il default.

Esempi di collocazione

Il criterio si chiarisce con tre esempi tipici.

- **Il contesto di un cliente specifico**. Si lavora per un'azienda con un proprio glossario di termini interni, persone identificate per ruolo, abitudini comunicative specifiche, ad esempio una preferenza per le sintesi numerate e una richiesta di evitare emoji. Nessuna di queste informazioni ha senso applicata ai lavori per altri clienti, e nessuna è una regola di metodo abbastanza generale da stare nelle generali. Il loro posto sono le istruzioni del progetto dedicato a quel cliente, dove formano il contesto operativo entro cui Claude legge le richieste e formula le risposte.
- **L'attivazione persistente di una skill o di un MCP**. Si gestisce un progetto editoriale in cui ogni testo prodotto deve passare per una skill di assistenza editoriale che applica le convenzioni redazionali della testata. La regola «*in questo progetto attiva sempre la skill di assistenza editoriale per la produzione di tutti i contenuti scritti*» nelle istruzioni di progetto risparmia di richiamarla a ogni chat. È una regola che ha senso solo dentro l'ambito del progetto editoriale, perché negli altri progetti la stessa skill non serve o serve in modo diverso. Per la portabilità vale quanto visto nella sezione *La stessa istruzione, ambienti diversi*. Una skill di account funziona in tutti gli ambienti in cui il progetto può essere aperto, una skill locale solo dentro Claude Desktop e solo sul computer in cui è installata.
- **Variazione del default delle generali per un contesto specifico**. Riprendendo l'esempio del capitolo 3 dal lato del progetto, si pensi a un progetto di brainstorming creativo in cui il rigore di verifica delle fonti che vale come default nelle generali ostacola la generazione di ipotesi. Una regola di progetto che allenta la verifica in favore della produzione di idee è la specializzazione operativa di una metodologia generale, scritta apposta per il contesto in cui quella variazione serve. La regola generale resta nelle

generali, perché continua a valere nella maggior parte dei lavori, e il progetto si limita a dire la differenza.

Un template di configurazione commentato

Il template che segue è pensato come proposta di partenza per le istruzioni di un progetto, non come modello da replicare. Il criterio di fondo è quello visto nella sezione *Criteri di collocazione*. Qui va solo ciò che è specifico del progetto e si discosta dal default delle istruzioni generali. Le sezioni sono indicative, possono essere fuse, divise o rinominate in base alla natura del progetto. Quello che conta è che ogni blocco esprima differenze, non default.

```
## Contesto del progetto

[Per chi è stato creato il progetto, ambito tematico, interlocutori abituali, tipo di
output atteso. Solo ciò che orienta Claude in modo specifico per questo lavoro, non un
riassunto del lavoro stesso.]

## Preferenze specifiche

[Lingua, registro, lunghezza, formato di risposta, scelte formali di scrittura, solo se
diverse da quelle delle istruzioni generali. Se una preferenza coincide con il default
non va ripetuta qui.]

## Metodo specifico per il progetto

[Eventuali deviazioni dal metodo di default delle istruzioni generali, ad esempio
l'allentamento del rigore di verifica per i progetti di brainstorming, o un rinforzo
specifico per i progetti di analisi.]

## Strumenti e risorse del progetto

[Skill, MCP, file e cartelle eventualmente presenti e pertinenti al progetto, da elencare
solo quando effettivamente caratterizzano il lavoro. Per le risorse locali, dichiarare in
apertura se il progetto è destinato all'uso in Claude Desktop e su quale computer.]
```

Le quattro sezioni rispecchiano la struttura del template per le istruzioni generali, viste al capitolo 3, con due differenze pensate apposta per il livello di progetto. La prima è l'assenza della sezione *Profilo*, che non si replica perché il profilo dell'utente è già nelle generali e qui sarebbe ridondanza. La seconda è la trasformazione della sezione *Strumenti e skill* in *Strumenti e risorse del progetto*, più specifica e con una dimensione di portabilità che al livello base non c'era.

La sezione *Contesto del progetto* è quella che caratterizza maggiormente il livello, e va riempita con misura. Va detto a Claude per chi sta lavorando, su cosa, e con che tipo di output, ma evitando il riassunto enciclopedico del progetto stesso. Le informazioni che servono sono quelle che orientano il comportamento, non quelle che descrivono in profondità il dominio. La documentazione di dominio è più efficace come materiale caricato nei file del progetto, dove Claude può consultarla quando serve, piuttosto che condensata nelle istruzioni dove rischia di diluire la priorità delle regole.

La sezione *Preferenze specifiche* va scritta tenendo presente la regola d'oro, ciò che coincide con il default non va qui. Se nelle istruzioni generali si è scelto l'italiano e il progetto resta in italiano, non si specifica l'uso dell'italiano nel progetto. Si scrive solo dove ci si discosta. La stessa logica vale per il registro, la lunghezza, le scelte formali. Una sezione asciutta che dica solo le differenze è più efficace di una sezione completa che ribadisce il default.

La sezione *Metodo specifico per il progetto* è la più importante e anche la più trascurata, perché è la più astratta. È il posto in cui si scrive non cosa Claude deve fare, ma come deve affrontare il lavoro. Per i progetti di analisi può rinforzare la verifica delle fonti, per i progetti di brainstorming può allentarla, per i progetti che richiedono particolare attenzione a certi tipi di errore può segnalarli esplicitamente. Andare oltre il default delle generali in questa sezione cambia il modo in cui Claude si pone, non solo cosa risponde.

La sezione *Strumenti e risorse del progetto* è il punto di intersezione più delicato con la portabilità. Ha senso solo quando il progetto si appoggia effettivamente a strumenti o risorse specifici, in molti progetti può restare vuota. Le skill di account, gli MCP disponibili in tutti gli ambienti, le indicazioni di metodo sull'uso degli strumenti, sono regole portabili e stanno bene qui. Le risorse locali, file su disco, skill locali, MCP non disponibili nell'interfaccia web, vanno qui solo nei progetti destinati all'uso in Claude Desktop. In quel caso conviene aprire la sezione con una dichiarazione esplicita del tipo «questo progetto è pensato per essere usato in Claude Desktop, su un computer che ha installato i seguenti elementi».

Anti-pattern ricorrenti

Quattro errori si ripetono con sufficiente frequenza da meritare una nota più estesa. Sono i comportamenti che, nei fatti, separano una configurazione di progetto efficace da una che produce risultati incoerenti senza che si capisca subito perché.

- **La duplicazione di regole già presenti nelle istruzioni generali**. Si scrive nel progetto una regola che esiste già al livello base, di solito perché si vuole essere sicuri che venga applicata, oppure perché più banalmente ci si è dimenticati d'averla già inserita nel livello base. Il risultato è che la stessa regola compare in due posti, e il prompt di sistema effettivo della chat la contiene due volte. Questo non rinforza la regola, anzi la indebolisce. L'asimmetria fra una formulazione articolata nelle generali e una formulazione abbreviata nel progetto, o viceversa, introduce un margine interpretativo che Claude deve risolvere da solo. Anche quando le due formulazioni sono identiche, la duplicazione è un costo di manutenzione. Qualsiasi aggiornamento futuro va replicato in entrambi i posti, e prima o poi si dimentica di farlo. Quando si è tentati di ripetere una regola per sicurezza, conviene resistere e fidarsi del meccanismo di stratificazione.
- **L'accumulo nelle istruzioni di un progetto di regole che in realtà varrebbero per tutti i progetti**. Si lavora a un progetto specifico, ci si accorge che una certa regola sarebbe utile, la si scrive lì. Mesi dopo si apre un altro progetto e la stessa regola, mai trasferita, non agisce più. Il progetto in cui la regola è stata scritta diventa una specie di prima approssimazione della propria configurazione generale, una bozza non ufficiale di ciò che dovrebbe stare nelle generali. La soluzione è una pratica di promozione. Ogni volta che si scrive una regola nelle istruzioni di un progetto vale la pena chiedersi se vale solo lì o

ovunque. Se la risposta è ovunque, la regola va spostata nelle generali, e la versione del progetto va rimossa o ridotta alle eventuali specializzazioni residue. Una revisione periodica delle istruzioni di progetto serve esattamente a questo, riconoscere le regole che hanno guadagnato pervasività nel tempo e promuoverle al livello giusto.

- **La confusione fra istruzione di progetto e richiesta puntuale**. Si scrive nel campo istruzioni del progetto qualcosa che è in realtà un compito specifico, una richiesta puntuale, un brief per una singola conversazione. Il sintomo tipico è un'istruzione che inizia con un imperativo molto concreto del tipo «scrivi un articolo su...» o «analizza il documento allegato e produci un riassunto». Sono richieste, non regole, e il loro posto è il prompt della singola chat. Quando finiscono nelle istruzioni di progetto, si applicano come contesto a tutte le chat successive del progetto, anche a quelle che non avrebbero ragione di tenerne conto, e tendono a invecchiare diventando rumore residuo. La soluzione è una distinzione di ruolo. Le istruzioni di progetto contengono regole pervasive che valgono per tutto il lavoro nel progetto, le richieste e i compiti puntuali stanno nei prompt delle singole chat. Una buona istruzione di progetto è scritta in modo da essere ancora pertinente fra sei mesi, una richiesta puntuale è scritta per essere consumata e dimenticata.
- **La dipendenza implicita dall'ambiente**. Si scrive nelle istruzioni di un progetto una regola che presuppone risorse locali, file in un percorso, una skill installata sul disco, un MCP disponibile solo in Desktop, senza dichiarare in apertura che il progetto è pensato per quell'ambiente. La regola funziona perfettamente fino a quando il progetto viene aperto altrove, sull'interfaccia web, su un altro computer, con una configurazione di MCP diversa. A quel punto fallisce silenziosamente. Silenziosamente è la parte importante, Claude non può dire «questa istruzione non vale qui», semplicemente non riesce a eseguirla, e l'utente vede comportamenti incoerenti senza capirne la causa. La soluzione è la dichiarazione esplicita di destinazione, già introdotta nella sezione *La stessa istruzione, ambienti diversi*. Quando un progetto ha dipendenze locali, la prima riga delle istruzioni dichiara dove il progetto è destinato a essere aperto e quali risorse devono essere disponibili. È un anti-pattern di disciplina, non di tecnica, e richiede di scrivere una riga in più ogni volta che si configurano dipendenze locali. È poco, e ripaga in chiarezza nel tempo.

In tutti e quattro i casi vale un'osservazione di metodo. La revisione periodica delle istruzioni di progetto, fatta a freddo a distanza di settimane dalla scrittura, è il modo più economico per intercettare questi errori prima che producano effetti.

Le istruzioni di Cowork

Le istruzioni di Cowork si applicano ovviamente all'area Cowork di Claude Desktop. Qui l'assistente esegue codice, opera sul filesystem, attiva skill e server MCP locali, e delega compiti ad agenti che operano in modo autonomo. Le istruzioni si configurano da *Impostazioni > Cowork*, in un campo di testo libero accompagnato dalla descrizione «*Le istruzioni qui si applicano a tutte le sessioni Cowork. Usa questo spazio per preferenze, convenzioni o contesto che Claude dovrebbe sempre conoscere.*».

Il capitolo indica cosa ha senso scrivere a questo livello e cosa lasciare altrove. Propone un template come proposta di partenza e raccoglie gli anti-pattern ricorrenti.

Quando vengono lette

Le istruzioni di Cowork vengono lette all'avvio di ogni sessione di lavoro nell'area Cowork di Claude Desktop, e si compongono nel prompt di sistema della conversazione insieme alle istruzioni generali. Da quel momento valgono per tutta la durata della sessione.

Nella configurazione sono da considerare anche due aspetti particolari.

```
Il primo riguarda l'uso su macchine diverse, le istruzioni di Cowork vivono nel
filesystem della singola macchina su cui Claude Desktop è installato e non si
sincronizzano fra computer diversi. Chi usa Claude Desktop su più pc può avere tante
configurazioni di Cowork quante sono le installazioni. Conviene decidere se le
configurazioni debbano essere allineate o possano divergere. Nel primo caso le si
aggiorna insieme quando se ne modifica una, replicando la stessa configurazione
manualmente. Nel secondo caso si configura ciascuna macchina secondo le sue esigenze, per
esempio se i due pc sono dedicati a usi diversi con strutture di cartelle differenti.
```

```
Il secondo riguarda i progetti Cowork, le cui chat ricevono un prompt di sistema composto
dalle istruzioni generali, da quelle di Cowork e da quelle del progetto. Se poi il
progetto Cowork viene creato partendo da un progetto Chat ed eredita quindi in sola
lettura le istruzioni del progetto Chat, il prompt di sistema in quel progetto Cowork
comprende anche le istruzioni del progetto Chat ereditato. Le istruzioni di Cowork
restano attive in tutte le sessioni Cowork, dentro o fuori da un progetto, e fanno da
fondamento a quelle di progetto.
```

Criteri di collocazione

Nelle istruzioni di Cowork vanno le regole che hanno senso solo in un ambiente in cui Claude esegue codice, opera sul filesystem, attiva skill e server MCP, delega ad agenti. Sono regole operative legate all'esecuzione locale, fuori da quel contesto sarebbero fuori posto.

Rientrano bene le convenzioni operative legate all'ambiente di esecuzione. Nomi e formati dei file, posizione delle cartelle di output, gestione dei backup prima di modificare file esistenti. Indicazioni sulla produzione di codice, dalla versione del linguaggio target alla gestione degli errori e allo stile dei commenti. Sono regole che presuppongono un ambiente in cui Claude effettivamente legge e scrive sul disco e talvolta esegue codice.

Rientrano bene le regole sull'uso delle skill e degli strumenti di ricerca. Quali skill consultare come prima fonte, come integrarle con altri strumenti, in che ordine usare i tool nativi rispetto agli MCP. Quando preferire un MCP locale a uno di account. Vale qui un'osservazione di portabilità inversa rispetto a quella vista nei capitoli precedenti. Le skill di account funzionano ovunque, le skill locali solo sulla macchina su cui sono installate. Una regola di Cowork che richiama una skill locale è coerente con il livello, perché Cowork è già di per sé locale.

Rientrano le regole sull'uso degli agenti. Quando delegare e quando eseguire direttamente, livello di autonomia degli agenti, momenti in cui Claude si ferma a chiedere conferma e ambiti in cui può procedere fino al completamento. Sono regole che hanno senso solo dove gli agenti sono disponibili e dove i compiti possono richiedere sequenze lunghe di passi senza supervisione continua. Conviene estendere agli agenti anche le regole più generali del livello, ad esempio sulle priorità fra strumenti. Un agente che applica criteri diversi da quelli del Claude principale produce risultati incoerenti. La regola si propaga lungo la catena, quando un agente a sua volta delega a sub-agenti i criteri vanno mantenuti su tutti i passaggi.

Una nota finale sulla pertinenza. Nelle istruzioni di Cowork la sintesi paga, ma non al prezzo di omettere convenzioni operative che caratterizzano davvero questo livello. Il punto di equilibrio sta nella pertinenza, ogni regola dovrebbe dire qualcosa di specifico per Cowork. Una configurazione che ribadisce regole già attive nelle generali o che anticipa specificità di singoli progetti diluisce le regole davvero utili. Il livello esiste per dire ciò che è specifico dell'ambiente Cowork.

Esempi di collocazione

Un esempio rende concreto cosa va a questo livello e riguarda la gestione degli agenti. Si lavora regolarmente con catene di delega in cui Claude assegna porzioni del lavoro ad agenti autonomi che a loro volta possono delegare a sub-agenti. Si vuole che le priorità fra strumenti decise nelle istruzioni di Cowork valgano lungo tutta la catena. La regola «*le priorità fra strumenti definite in queste istruzioni si applicano anche agli agenti delegati e ai loro sub-agenti*» propaga il criterio lungo l'esecuzione. Senza una regola esplicita di estensione, gli agenti possono scegliere strumenti diversi da quelli del Claude principale, e sui lavori lunghi questo produce risultati incoerenti.

Un template di configurazione commentato

Il template che segue è pensato come proposta di partenza per le istruzioni di Cowork, non come modello da replicare. Il criterio di fondo è quello visto nella sezione *Criteri di collocazione*. Qui va solo ciò che è specifico dell'ambiente Cowork. Le sezioni sono indicative, possono essere fuse, divise o rinominate in base al proprio modo di lavorare.

```
## File e output

[Convenzioni sui nomi dei file, formati di salvataggio, posizione
delle cartelle di output, gestione dei backup prima di modificare
file esistenti. Indicazioni che presuppongono accesso effettivo
al filesystem.]

## Skill e strumenti

[Quali skill consultare come prima fonte, in che ordine, come
integrarle con altri strumenti. Priorità fra tool nativi e server
MCP, preferenze fra MCP locali e di account. Solo le indicazioni
che valgono per tutto il lavoro in Cowork.]

## Produzione di codice

[Versione del linguaggio target, gestione degli errori di base,
stile dei commenti. Convenzioni di formato e organizzazione del
codice prodotto.]

## Agenti

[Quando delegare compiti ad agenti, livello di autonomia,
momenti di verifica obbligatoria, regole generali del livello
da estendere agli agenti e da mantenere lungo la catena
quando un agente delega a sub-agenti.]

## Workflow operativi (opzionale)

[Procedure ricorrenti che riguardano l'intero ambiente Cowork.
Sequenze tipiche di passi per gestire una categoria di compiti,
verifiche obbligatorie prima di certe azioni.]
```

Le cinque sezioni rispecchiano le categorie di regole tipiche di Cowork. Non c'è una sezione *Profilo* né una sezione *Preferenze di default*, perché quei contenuti vivono al livello base. Qui si scrive solo dove ci si discosta per ragioni legate all'ambiente.

La sezione *File e output* è quella che caratterizza maggiormente il livello. Tutte le regole su nomi, formati, posizione e backup hanno senso solo dove Claude tocca davvero il filesystem. Va riempita con cura, è il punto in cui la configurazione produce gli effetti più visibili nel lavoro quotidiano. Va riempita anche con misura, evitando di anticipare convenzioni che varrebbero solo per certi progetti.

La sezione *Skill e strumenti* è la più delicata sul piano della portabilità tra macchine. Una regola che richiama una skill di account funziona ovunque. Una regola che richiama una skill locale funziona solo sulla macchina su cui la skill è installata. Per Cowork, che è già di per sé locale, una regola che presuppone una skill locale è coerente con il livello. Diventa un problema solo se la stessa skill non è installata sugli altri pc su cui Cowork viene usato e si vuole importare la stessa configurazione. In quel caso la regola va vincolata alla configurazione effettiva di ciascuna macchina.

La sezione *Produzione di codice* va tenuta misurata. Le indicazioni di stile producono effetti se Cowork viene usato regolarmente per scrivere codice, hanno meno valore se l'attività di codifica è occasionale.

Conviene scriverle solo dopo essere arrivati a una preferenza stabile, altrimenti il rischio è di vincolare Claude a convenzioni che si vorranno cambiare presto.

La sezione *Agenti* è la più tipica di Cowork. Le regole qui dicono dove e quando delegare, con quale autonomia, e come trasferire agli agenti i criteri del livello. Una configurazione che non si esprime sugli agenti lascia il comportamento al default di Claude. Sui lavori lunghi questo può produrre risultati incoerenti, perché un agente può scegliere criteri diversi da quelli del Claude principale. L'effetto si amplifica nelle catene a più livelli, dove ogni passaggio può aggiungere una propria deviazione.

La sezione *Workflow operativi* è opzionale. Va affrontata solo se ci sono procedure ricorrenti che caratterizzano davvero il modo di lavorare in Cowork. Una sequenza di passi che si ripete ogni volta che si produce una certa tipologia di file, una verifica obbligatoria prima di certe azioni, un protocollo di salvataggio. Si distingue dalle altre sezioni perché riguarda il *quando* e il *come ordinare* le azioni, mentre le altre riguardano i *contenuti* di ciascuna azione.

Anti-pattern ricorrenti

Quattro errori si ripetono con sufficiente frequenza da meritare una nota più estesa.

- Il primo è l'inserimento nelle istruzioni di Cowork di regole che varrebbero ovunque. Si scrive in Cowork una preferenza di metodo, una convenzione linguistica, una regola di verifica delle fonti. La si formula mentre si lavora in Cowork, e Cowork è il campo che si ha sotto gli occhi. Il risultato è che la regola si applica solo nelle sessioni Cowork. Negli altri ambienti non viene letta, anche se varrebbe altrettanto bene anche lì. Il rimedio è promuovere la regola al livello giusto, le istruzioni generali, lasciando in Cowork solo l'eventuale specializzazione legata all'ambiente. La pratica della promozione e della retrocessione delle regole è trattata in modo sistematico al capitolo 7.
- Il secondo è speculare al primo. È l'accumulo nelle istruzioni di Cowork di specificità che riguardano singoli progetti Cowork. Si lavora a un progetto specifico, si individua una convenzione utile, la si scrive in Cowork pensando «così la trovo sempre». Il problema è che la convenzione si applica anche alle sessioni di altri progetti dove non c'entra nulla. Lì compare come rumore residuo che orienta il comportamento di Claude in modo inappropriato. La soluzione è la disciplina della collocazione, ogni regola che riguarda un singolo progetto Cowork va nel campo istruzioni di quel progetto. Lì vive nel suo ambito naturale e non interferisce con il resto.
- Il terzo è la dipendenza implicita dalla macchina, combinata con l'aspettativa di portabilità. Si scrive in Cowork una regola che richiama un percorso del filesystem o una skill locale. Cowork è già di per sé locale e la dipendenza è coerente con il livello. Fin qui nessun problema. Il problema nasce quando Claude Desktop viene installato su un secondo pc. Le istruzioni di Cowork del secondo pc partono vuote. La stessa regola, replicata o no, si scontra con un percorso diverso o una skill non installata. La regola in sé non era sbagliata, sbagliata era l'ipotesi implicita che la configurazione fosse la stessa ovunque. La soluzione è duplice. Dichiarare nella configurazione di un pc le dipendenze locali presupposte, in modo che il lettore-autore le veda quando rilegge la sua configurazione. Decidere consapevolmente se le configurazioni di pc diversi vanno tenute allineate o lasciate divergere, e in entrambi i casi gestirle con una procedura di manutenzione coerente.

- Il quarto è la configurazione che si esprime sugli agenti senza estendersi alla catena. Si scrive una regola operativa per Claude, ad esempio sulle priorità fra strumenti. La si formula come se valesse solo per il Claude principale che riceve direttamente la richiesta dell'utente. Quando il Claude principale delega a un agente, la regola può essere interpretata come riferita solo al primo. La formulazione tipica parla di «Claude» o di «il sistema», senza specificare chi è incluso quando ci sono catene di delega. Sui lavori lunghi che richiedono catene di delega, l'effetto è una progressiva deriva dalle regole impostate. La soluzione è l'estensione esplicita. Formulazioni come «*questa priorità vale anche per i sub-agenti*» propagano la regola lungo tutta l'esecuzione. Senza un'estensione esplicita, la regola vincola solo il primo passo.

Una rilettura periodica delle istruzioni di Cowork aiuta a intercettare questi errori sul nascere. Vale soprattutto perché alcuni di essi maturano nel tempo, man mano che si aggiungono regole senza rivedere quelle già presenti.

Gli stili di scrittura

Gli stili di scrittura sono trasversali agli altri livelli e dedicati alla forma delle risposte, agendo su tono, registro, formato e struttura. Si attivano e disattivano per singola chat, anche a metà conversazione. Si selezionano dal menu *Usa stile*, accessibile dal pulsante *+* in basso a sinistra dell'area di chat.

Sono gestibili solo nell'area Chat di claude.ai, nella Chat di Claude Desktop e nell'app per smartphone. Lo stile attivato nella Chat di Claude Desktop si propaga anche alle sessioni Cowork dello stesso computer, secondo un meccanismo descritto nella sezione *Quando si applicano*. La granularità per chat e la trasversalità sono le due caratteristiche che distinguono il loro livello dagli altri, ed è da queste due caratteristiche che discende il criterio per scriverli bene.

Gli stili sono di due tipi. Esistono cinque versioni predefinite, *Normale*, *Apprendimento*, *Conciso*, *Esplicativo* e *Formale*, ciascuna con un comportamento specifico. Accanto a questi ci sono anche gli stili personalizzati creati dall'utente, costruiti a partire da un esempio caricato o da una descrizione diretta. Le due famiglie convivono nel menu di selezione, e la sezione *Stili predefiniti e stili personalizzati* entra nel dettaglio di entrambe.

Il capitolo descrive il funzionamento degli stili e indica con quale criterio scrivere uno stile personalizzato. Propone un template come proposta di partenza e raccoglie gli anti-pattern ricorrenti.

Quando si applicano

Gli stili di scrittura sono l'unico livello di configurazione gestito esplicitamente dall'utente. Gli altri tre livelli vengono letti automaticamente all'avvio della conversazione, lo stile invece richiede una selezione dal menu *Usa stile*. In assenza di selezioni diverse il livello opera con *Normale*, lo stile predefinito che corrisponde al comportamento di default di Claude e che il menu mostra come attivo finché non si sceglie altro. Quando uno stile diverso da *Normale* è attivo compare un'icona a forma di penna nella barra del prompt accanto al pulsante *+*, come segnale visivo.

Lo stile selezionato si applica alla risposta successiva e a tutte quelle che la seguono. Si può cambiare in qualsiasi momento durante la conversazione, anche dopo aver già ricevuto risposte in uno stile diverso. Il cambio ha effetto sui nuovi messaggi e sugli interventi di modifica o ripetizione di una richiesta precedente. I messaggi già ricevuti restano scritti nello stile attivo al momento della loro generazione. È una caratteristica peculiare degli stili, che gli altri livelli non hanno, dove una modifica entra in vigore solo dalle conversazioni successive.

Lo stile selezionato resta attivo anche nelle conversazioni successive che si aprono nello stesso ambiente, finché l'utente non lo cambia. La persistenza è per ambiente, non globale per l'account. Uno stile scelto in Claude Desktop vale in Desktop, su claude.ai vale lo stile selezionato lì, e le scelte di claude.ai e Desktop

non si trasportano l'una nell'altra. In Desktop fra Chat e Cowork esiste invece un meccanismo di propagazione che il paragrafo successivo descrive.

In Desktop, fra la Chat e l'area Cowork dello stesso computer, la propagazione opera con regole specifiche. Al momento dell'apertura di una sessione Cowork viene ereditato lo stile che è attivo nella Chat, restando operativo per tutta la durata della conversazione. Nell'interfaccia di Cowork compare l'icona dello stile, ma non c'è un menu per gestirlo. L'unica azione possibile è cliccare la **X** dell'icona, che fa sparire la segnalazione e propaga la disabilitazione alla Chat, che torna a *Normale*. Lo stile della sessione Cowork in corso resta però quello letto all'apertura, indipendentemente da modifiche fatte successivamente, sia cambiando stile in Chat sia intervenendo sull'icona in Cowork. Le modifiche allo stile vengono ereditate solo dalle sessioni Cowork avviate successivamente. Il comportamento è osservato in modo riproducibile, non è descritto nella documentazione di Anthropic. In quanto comportamento non documentato, potrebbe essere modificato in versioni future di Claude Desktop senza preavviso.

```
Sulla disponibilità del livello, gli stili predefiniti sono accessibili in tutti e tre
gli ambienti che hanno l'area Chat (web, mobile e Desktop), più Cowork con le modalità
descritte. La creazione e la modifica di stili personalizzati sono disponibili solo in
claude.ai e nella Chat di Claude Desktop, mentre sull'app per smartphone si può solo
selezionare uno stile fra quelli già disponibili. Una conseguenza pratica è che chi vuole
impostare un proprio stile personalizzato lo deve fare dall'interfaccia desktop o web. Lo
ritrova poi selezionabile anche da mobile per le conversazioni in mobilità.
```

Criteri di collocazione

Rientrano bene le preferenze di forma che hanno senso solo in alcuni momenti del lavoro. Uno stile sintetico per le risposte brevi e dirette, uno stile dettagliato per gli approfondimenti, uno stile in registro alto per i documenti destinati a terzi. Sono casi in cui la regola serve in alcune chat e non in altre, e cambia in modo naturale a seconda del compito.

Rientrano bene le preferenze che attraversano progetti diversi senza appartenere a nessuno in particolare. Uno stile orientato all'esplorazione di idee, attivabile indistintamente in vari progetti dove si vogliono sessioni di brainstorming. Uno stile per la comunicazione formale a destinatari esterni, usato in contesti diversi accomunati solo dalla destinazione del testo. Sono regole che vivono trasversalmente all'ambito dei progetti, e qui trovano una collocazione che nessun altro livello offre.

Rientrano bene le scelte di vocabolario, ritmo, struttura che variano per tipo di output. Una preferenza per le frasi brevi quando si scrivono comunicazioni interne, una preferenza per il periodare ipotattico quando si scrivono testi argomentativi. Sono regole sulla formulazione che la granularità del livello permette di attivare e disattivare a seconda del compito specifico.

Una nota finale sulla pulizia. Uno stile che porta in modo coerente la propria caratteristica, granularità e trasversalità, è più affidabile di uno stile che cerca di fare tutto. Il livello esiste per dire ciò che è specifico della formulazione granulare, e che ha senso poter cambiare al volo. La coerenza con questa caratteristica è il punto di equilibrio che orienta le scelte di scrittura.

Esempi di collocazione

Il livello si chiarisce con tre esempi tipici.

- Il primo riguarda una variazione granulare di approfondimento. Si lavora a varie tipologie di compiti, alcuni richiedono risposte estese e articolate, altri risposte sintetiche di poche righe. Il default delle istruzioni generali, scelto per la maggior parte dei lavori, è un livello di approfondimento medio. Per i casi in cui si vogliono risposte sintetiche, si attiva *Conciso*. Per quelli che richiedono spiegazioni più estese, si attiva *Esplicativo*. La preferenza varia da chat a chat, talvolta anche a metà di una stessa chat, e attraversa indifferentemente i progetti. Lo stile è il livello in cui questa variabilità trova il suo posto.
- Il secondo riguarda la combinazione su livelli diversi. Le istruzioni generali chiedono italiano, frasi medie, prosa narrativa. Lo stile *Formale* è attivo per una conversazione in cui si sta lavorando a una proposta professionale a un cliente. La risposta arriva in italiano con frasi medie e in stile narrativo, con il registro formale aggiunto dallo stile. Le indicazioni dei due livelli si combinano senza concorrenza, ognuno opera sul piano che gli compete, lingua e prosa al livello base, registro al livello stile. Quando la conversazione finisce, lo stile può essere disattivato o lasciato attivo per altre conversazioni dello stesso tipo, senza che le scelte dei livelli sottostanti vengano toccate.
- Il terzo riguarda la trasversalità ai progetti. Si lavora abitualmente su diversi progetti, alcuni di consulenza, altri di formazione, altri di scrittura editoriale. In ognuno di questi progetti capita di aprire sessioni esplorative. Si vuole un Claude meno orientato alla risposta diretta e più pronto a fare domande, sfidare le assunzioni, suggerire prospettive alternative. Il bisogno attraversa l'ambito dei progetti, è lo stesso quando si è in consulenza, in formazione o in scrittura. Uno stile personalizzato *Brainstorming* attivabile in qualsiasi progetto è la collocazione che sfrutta esattamente la caratteristica trasversale degli stili. Quando la sessione esplorativa finisce, si torna a *Normale* e le scelte di default tornano a operare.

I tre esempi seguono lo stesso schema. Negli stili va ciò che è legato alla granularità per chat o alla trasversalità ai progetti, sfruttando l'una, l'altra o entrambe. La presenza di queste due caratteristiche nella regola è il segnale che si è trovato il livello giusto.

Stili predefiniti e stili personalizzati

Gli stili di Claude sono di due tipi, predefiniti e personalizzati. I cinque predefiniti coprono le esigenze più comuni e si attivano direttamente dal menu *Usa stile*, senza dover passare per la creazione. Gli stili personalizzati permettono invece di costruire una propria configurazione di forma, partendo da zero. Si gestiscono dal popup *Personalizza i tuoi stili*, accessibile dalla voce *Crea e modifica stili* dello stesso menu.

I cinque predefiniti sono *Normale*, *Apprendimento*, *Conciso*, *Esplicativo* e *Formale*. Ciascuno ha un comportamento specifico.

- *Normale* è quello attivo all'apertura di una conversazione e corrisponde al comportamento di default di Claude.
- *Conciso* produce risposte brevi e dirette, senza preamboli e con minore strutturazione interna, adatto alle richieste in cui si vuole arrivare rapidamente al punto.
- *Esplicativo* fa il contrario, allunga le risposte, aggiunge contesto e spiegazioni di sfondo, adatto a chi vuole capire un concetto a fondo.
- *Formale* sposta il registro verso un tono professionale, asciuga le contrazioni, organizza meglio la struttura interna, adatto a documenti, comunicazioni di lavoro e materiale destinato a terzi.
- *Apprendimento* introduce un approccio socratico, in cui Claude pone domande, sfida le assunzioni e guida l'utente a costruire la risposta da sé invece di fornirla direttamente.

Normale ha una natura particolare rispetto agli altri quattro predefiniti. Non aggiunge un proprio set di istruzioni di stile sopra i livelli sottostanti, ma corrisponde all'assenza di un'indicazione esplicita di stile. Lascia operare quanto eventualmente definito nelle istruzioni generali, di progetto o di Cowork sul piano della formulazione. È il motivo per cui le istruzioni di stile scritte negli altri livelli sono effettivamente attive quando il menu mostra *Normale* come selezione corrente. La documentazione di Anthropic non descrive esplicitamente questo comportamento, ma è coerente con la logica del prompt di sistema che governa l'applicazione degli stili e con il comportamento osservato.

Quando i cinque predefiniti non sono sufficienti si crea uno stile personalizzato. I cinque predefiniti non sono modificabili dall'utente, lo stile personalizzato si costruisce da zero. Le vie disponibili sono due, complementari ma alternative al momento della creazione. La prima è fornire un esempio di scrittura caricato in formato pdf, doc o txt, oppure incollato come testo. La seconda è scrivere direttamente le istruzioni di stile, opzione segnalata nell'interfaccia come *Descrivi lo stile invece*.

La via dell'esempio caricato è più adatta quando si ha a disposizione un testo proprio già scritto nel registro che si vuole replicare. Articoli pubblicati, documenti aziendali, comunicazioni a clienti sono tutti materiali che funzionano bene come riferimento. Claude analizza tono, struttura e vocabolario dell'esempio e ne ricava una propria sintesi, da cui costruisce lo stile. La qualità del risultato dipende dalla coerenza e dalla rappresentatività del testo caricato. Un esempio breve o eterogeneo produce uno stile poco caratterizzato, un esempio lungo e omogeneo produce uno stile più riconoscibile.

La via delle istruzioni dirette è più adatta quando si ha in mente un'idea precisa dello stile desiderato, senza un testo già scritto che la rappresenti. Vale anche quando si vuole il massimo controllo descrittivo. Si scrive un testo che descrive il registro, il tono, la struttura, le scelte tipografiche desiderate. È la via più precisa, ma richiede di formulare con cura le istruzioni. Lo stile risulta tanto definito quanto è definito il testo che lo descrive.

Come si combinano con gli altri livelli

Gli stili non operano in isolamento. Vivono dentro un sistema dove tre livelli sottostanti, generali, di progetto e di Cowork, hanno potenzialmente già trattato la formulazione delle risposte. La scelta di scrivere qualcosa

nello stile dipende da cosa è già definito sotto, perché altrimenti si rischia di duplicare o di sopravanzare in modo non consapevole.

Le interazioni possibili fra lo stile e i livelli sottostanti sono tre.

- **Il completamento**. Lo stile dice qualcosa sulle dimensioni della formulazione che i livelli sottostanti lasciano libere. Le istruzioni di progetto fissano lingua e registro, lo stile aggiunge la lunghezza target o le convenzioni tipografiche. È il modo più pulito di usare il livello, perché ogni elemento dice qualcosa di nuovo e la combinazione al momento della lettura del prompt è lineare.
- **Il rafforzamento**. Lo stile prende una dimensione già toccata nei livelli sottostanti e la rende più dettagliata, specializzandola per il tipo di output a cui lo stile è dedicato. Le istruzioni generali indicano «*scrivere in stile divulgativo professionale*», lo stile *Articolo blog* specifica per quel tipo di output che le frasi stanno fra le 15 e le 25 parole, che la prosa è narrativa senza elenchi puntati, che le virgolette sono caporali e che il registro è quello fra colleghi competenti. La regola sotto resta valida, lo stile la rende operativamente più stringente nel contesto in cui si attiva. Va distinto dalla ripetizione, dove si trascrivono nello stile le stesse direzioni già presenti nei livelli sottostanti, senza aggiungere precisione, e si produce solo ridondanza. Il primo anti-pattern del capitolo entra nel dettaglio di questo errore.
- **L'alternativa**. Lo stile dice qualcosa di diverso da quanto fissato sotto. La regola di prevalenza vista al capitolo 2 fa vincere il più specifico, e con lo stile attivo la formulazione di default viene scavalcata per quella conversazione. È un uso legittimo, con un rischio da tenere presente. Quando lo stile resta attivo per settimane, è facile dimenticare quale regola era stata scritta nelle generali e percepire come comportamento errato ciò che è invece coerente con la stratificazione.

Il ruolo effettivo del livello stili nella configurazione di un utente dipende anche da come è organizzato il lavoro. Chi struttura il proprio lavoro per progetti densi tende a esaurire la formulazione nelle istruzioni di progetto. Per questo profilo gli stili hanno un uso ristretto, di solito alle variazioni granulari interne a un progetto, come l'alternanza fra *Conciso* ed *Esplicativo* a seconda del compito del momento. Chi lavora prevalentemente in conversazioni fuori dai progetti, dove sopra c'è solo il livello base, trova negli stili lo strumento principale per specializzare la formulazione su tipi di output ricorrenti, e li usa più spesso e con maggiore caratterizzazione.

Ci sono anche casi in cui conviene non usare uno stile. Quando la formulazione voluta coincide con il default impostato nei livelli sottostanti, lasciare *Normale* e far operare quanto già scritto è la scelta più pulita. Quando l'esigenza di variazione è puntuale, valida per una sola conversazione, conviene un'istruzione esplicita nel primo messaggio della chat. Uno stile apposito sarebbe sproporzionato a un uso unico. Quando la regola riguarda dimensioni che non sono di formulazione, come la lingua, le metodologie o l'attivazione di skill, vale il secondo anti-pattern del capitolo, lo stile non è il livello adatto.

La distribuzione delle regole sui livelli e l'orchestrazione complessiva della propria configurazione sono trattate in modo sistematico nel capitolo successivo.

Un template di stile commentato

Il template che segue è pensato come proposta di partenza per scrivere uno stile personalizzato in modalità *Descrivi lo stile invece*, non come modello da replicare. Il criterio di fondo è quello visto nella sezione *Criteri di collocazione*. Qui va solo ciò che riguarda la formulazione delle risposte e ha senso nella granularità del livello. Le sezioni sono indicative, possono essere fuse, divise o rinominate in base allo scopo dello stile.

```
## Registro e tono

[Livello di formalità o di prossimità con il lettore, presenza
o assenza di humor, distanza emotiva, eventuale neutralità o
calore della voce. Solo gli aspetti che vogliono variare
rispetto al default impostato nelle istruzioni generali o di
progetto.]

## Formato e struttura

[Prosa o liste, paragrafi corti o lunghi, presenza di titoli
intermedi, struttura introduttiva e conclusiva, uso di esempi
inline o separati. Indicazioni sulla forma del testo nella
risposta, non sul contenuto.]

## Lunghezza e ritmo

[Lunghezza target delle risposte, sintesi o estensione,
periodare paratattico o ipotattico, frasi brevi o lunghe,
varietà delle aperture. Il tipo di ritmo che si vuole
percepire nel testo finito.]

## Convenzioni tipografiche

[Uso del grassetto, del corsivo, delle virgolette, di
apostrofi e accenti. Eventuali preferenze su elenchi,
note, citazioni. Tutto ciò che riguarda la veste grafica
del testo.]
```

Le quattro sezioni coprono i piani principali su cui uno stile interviene tipicamente. Non c'è una sezione *Profilo* né una sezione *Metodo*, perché identità dell'utente e metodologie di lavoro vivono ai livelli più stabili. Non ci sono regole sull'attivazione di skill, sui server MCP o sulle convenzioni di file, perché lo stile interviene sulla formulazione delle risposte, non sull'apparato operativo.

La sezione *Registro e tono* è quella più caratterizzante. Distingue uno stile in modo netto, perché il registro è la scelta che il lettore percepisce per prima. Va riempita con misura, evitando descrizioni generiche del tipo «*professionale ma accessibile*», che lasciano troppo spazio interpretativo. Funziona meglio una descrizione che usa esempi concreti e contrasti, ad esempio «*come una conversazione fra colleghi competenti, mai didascalico, mai marketing speak*».

La sezione *Formato e struttura* incide sulla scansione visiva del testo. Il punto delicato è bilanciare struttura e fluidità. Una struttura troppo articolata trasforma ogni risposta in un documento, una struttura troppo

libera produce muri di testo difficili da leggere. Conviene esplicitare il punto di equilibrio desiderato, ad esempio la preferenza per la prosa con qualche elenco solo dove serve davvero, o viceversa.

La sezione *Lunghezza e ritmo* è la più sottile. Indicare una lunghezza target, ad esempio «*tra 150 e 300 parole salvo richieste più estese*», dà un'ancora concreta. Indicare un ritmo, ad esempio «*frasi brevi e medie alternate, evitare le subordinate concatenate*», agisce sulla cadenza della lettura. Le due indicazioni si rinforzano a vicenda, una lunghezza senza ritmo può produrre testi sintatticamente affaticati, un ritmo senza lunghezza tende a sforare.

La sezione *Convenzioni tipografiche* è quella che produce gli effetti più visibili nell'immediato. Le scelte tipiche sono il corsivo invece del grassetto, le virgolette caporali invece di quelle alte, la scelta di non usare elenchi puntati nei testi narrativi. Sono indicazioni che si vedono dalla prima risposta. Vale la pena scriverle in modo univoco, evitando formulazioni a doppia interpretazione, perché altrimenti il modello sceglie di volta in volta.

Anti-pattern ricorrenti

Cinque pattern ricorrono con regolarità nell'uso degli stili.

- **Replicare nello stile regole già presenti nelle istruzioni generali o di progetto**. Se nelle generali si è scritto «*scrivere in italiano in registro colloquiale fra colleghi*» e poi nello stile si ripete la stessa indicazione, si crea ridondanza inutile. Lo stile dovrebbe contenere solo ciò che si vuole far variare rispetto al default impostato altrove. Quando i due livelli vanno d'accordo, basta scrivere una volta sola al livello più stabile. Quando i due livelli divergono, è la divergenza a dover stare nello stile.
- **Scrivere nello stile regole che non riguardano la formulazione**. Attivazione di skill, metodologia di verifica, convenzioni sui file generati. Lo stile non è il livello giusto per queste indicazioni, perché ha granularità troppo fine. Se domani si attiva uno stile diverso, o si torna al default, quelle regole spariscono. La persistenza che queste regole richiedono è esattamente l'opposto della granularità per chat che caratterizza gli stili.
- **Descrivere lo stile in termini troppo generici**. Formule come «*professionale ma accessibile*», «*chiaro e diretto*», «*né troppo formale né troppo informale*» non vincolano niente. Il modello le interpreta in modo diverso volta per volta. Funzionano meglio le descrizioni con esempi concreti e contrasti, oppure le indicazioni operative, come «*frasi tra le 15 e le 25 parole*», «*niente elenchi puntati salvo dove servono davvero*», «*virgolette caporali per le citazioni*».
- **Creare uno stile per ogni progetto, ogni cliente, ogni compito**. La collezione cresce in fretta, il menu diventa ingombro, e la fatica di scegliere ogni volta supera il beneficio. Conviene tenere il numero di stili basso, due o tre, ciascuno usato di frequente. Per le esigenze occasionali è di solito più efficace una nota nel prompt o un'istruzione di progetto. Meglio non appesantire il livello stile con dettagli che si useranno una volta sola.
- **Dimenticare che lo stile resta attivo quando cambia il contesto, all'interno dello stesso ambiente**. Uno stile *Formale* impostato per scrivere una proposta a un cliente rimane anche nelle conversazioni

successive, anche su lavori di natura diversa. È utile prendere l'abitudine di controllare il menu *Usa stile* a inizio chat, e tornare a *Normale* o all'impostazione di default quando si cambia tipo di lavoro.

Una rilettura periodica del menu *Usa stile* aiuta a riconoscere gli stili che si usano davvero e a eliminare quelli che si sono accumulati senza ritrovare un uso stabile. La pratica della revisione della propria configurazione, valida per tutti i livelli, è trattata in modo sistematico al capitolo 7.

La collaborazione fra i livelli

I capitoli 3-6 hanno spiegato cosa contiene ciascuna configurazione, per ognuna hanno risposto alla domanda «qui cosa ci scrivo?». Questo capitolo guarda la configurazione nel suo insieme, non più una sua parte alla volta. La stessa regola scritta in parti diverse funziona in modi diversi, perché cambia l'ambito d'applicazione. Dove mettere una regola è una scelta che riguarda l'organizzazione complessiva, non solo la sua scrittura. Per capire il punto in cui ha senso inserirla bisogna partire dall'ambito d'uso di una regola, ad esempio l'indicazione sull'uso di una certa terminologia tecnica può stare nella configurazione generale, se vale in tutti i lavori. Oppure in quella di un progetto, se vale solo per un certo cliente. Una regola sul metodo di verifica delle fonti può stare nella generale, se è il modo abituale di lavorare. Oppure in quella di un progetto editoriale, dove la verifica delle fonti è centrale. La regola scritta è la stessa, ma a seconda di dove finisce vale per tutto o solo per un caso.

Il capitolo ha 5 sezioni. La prima descrive i tre modi in cui due regole presenti in livelli diversi possono interagire tra loro. La seconda mostra com'è fatta tipicamente una configurazione per i modi di lavorare più comuni. La terza spiega come la configurazione cambia nel tempo. La quarta tratta i casi in cui la stessa regola si trova scritta in più di un posto, e come gestirli. La quinta chiude con uno o due esempi completi.

Come si combinano le configurazioni

I tre modi descritti in questa sezione sono già stati introdotti al capitolo 6 a proposito di come gli stili si combinano con i livelli sottostanti. Qui vengono ripresi in forma generale, valida per qualsiasi coppia di livelli.

Regole presenti in punti diversi possono combinarsi in tre modi.

Il primo è il completamento. Due regole sullo stesso argomento dicono cose diverse, su aspetti diversi e i contenuti si sommano. Prendendo l'esempio di un utente specializzato in un certo ambito, nella configurazione generale dice di usare l'italiano con le modalità di scrittura di quel settore, in quella di un progetto editoriale aggiunge la terminologia tecnica e le convenzioni delle pubblicazioni scientifiche sempre di quel settore. Non c'è sovrapposizione, le due indicazioni riguardano aspetti che coesistono. È il caso più semplice, perché non c'è competizione.

Il secondo è il rafforzamento. Una regola riprende un'indicazione già data altrove e la specializza per un caso particolare. Ad esempio, la configurazione generale indica «registro divulgativo professionale»; in un progetto di formazione aziendale per ingegneri si specializza in «registro divulgativo fra colleghi tecnici esperti, esempi tratti dall'ambito industriale, niente paragoni dalla vita quotidiana». La regola generale resta valida, quella specifica la rende più precisa per quel contesto. È diverso dal copiare la stessa frase in due istruzioni. Ripetere la stessa indicazione non rafforza nulla, produce solo ridondanza.

Il terzo è l'alternativa. Una regola dice qualcosa di diverso da un'altra scritta altrove, e la regola di prevalenza vista al capitolo 2 fa vincere la più specifica. Ad esempio, la configurazione generale dice «rispondere sempre in italiano»; in un progetto si imposta invece «rispondere in inglese, il cliente lavora a Boston». Per quel progetto vince l'inglese, fuori dal progetto resta l'italiano. È un uso legittimo del meccanismo, con un rischio da tenere presente. Quando un'alternativa è impostata e resta attiva per mesi, è facile dimenticare la regola di partenza scritta nella generale. Il comportamento di Claude appare inatteso, ma è coerente con quanto è stato scritto.

Questi tre modi tornano nelle sezioni che seguono.

Configurazioni globali per profilo

Una configurazione complessiva è fatta in modo diverso a seconda di chi la costruisce. Cambia quali livelli si usano, cambia quante regole si mettono in ciascuno, cambia dove si concentra la maggior parte del lavoro. Non è una scelta fatta livello per livello in modo separato, è una scelta che si fa nel complesso, perché dipende dal modo in cui si lavora con Claude. A seconda del modo di lavorare può anche essere corretto non usare uno dei livelli.

Di seguito tre profili di configurazione, presentati come ipotesi rappresentative. Chi struttura il proprio lavoro intorno a progetti continuativi, chi lavora con agenti Cowork su Desktop, chi usa Claude in conversazioni isolate. La sezione descrive la configurazione tipica di ciascuno. Un paragrafo di chiusura tratta il profilo misto, che combina elementi dei tre.

Lavoro per progetti

Rientra in questo profilo chi struttura in modo continuativo il proprio lavoro intorno a progetti.

- Un consulente con più clienti, ognuno con un proprio progetto.
- Un formatore aziendale con più aziende seguite in parallelo.
- Un ricercatore con filoni di indagine distinti, ognuno con materiali, terminologia, output suoi.
- Un editore con più collane in lavorazione.

I casi sono diversi, ma il principio è lo stesso. Il progetto è il modo principale di organizzare il lavoro, e ogni progetto ha un suo contesto che dura nel tempo.

Le istruzioni di progetto contengono la maggior parte del lavoro di configurazione, e per questo sono dense e specifiche. Ogni progetto ha le sue regole su materiali di lavoro, terminologia tecnica del settore, modalità di output, vincoli editoriali, eventuali specificità del cliente o dell'oggetto. Possono essere lunghe, articolate in più sezioni, e valgono soltanto per quel progetto.

Le istruzioni generali, in questo profilo, sono tendenzialmente leggere, contenendo solo le regole davvero trasversali, quelle che attraversano tutti i progetti. La lingua, il registro di base, qualche convenzione di

metodo che resta valida ovunque. Non contengono dettagli sui temi di lavoro né riferimenti a clienti, perché ogni progetto si fa carico del suo contesto.

Cowork può comparire in coppia con i progetti per chi usa Desktop e gli agenti. Le istruzioni Cowork specializzano un agente per il tipo di lavoro del progetto, e si combinano con l'istruzione di progetto. Quando il lavoro con gli agenti è centrale, si entra nel profilo successivo.

Gli stili restano marginali in questo profilo, perché il progetto già definisce il registro di output. Possono tornare utili per piccole regolazioni dentro un progetto, come l'alternanza fra *Conciso* ed *Esplicativo* a seconda del compito, ma non sono al centro della configurazione.

È la scelta tipica di chi ha trovato nel meccanismo dei progetti il modo di tenere separati i contesti di lavoro, senza dover appesantire le generali con regole che valgono solo per alcuni casi.

Lavoro con agenti Cowork

Rientra in questo profilo chi usa Claude Desktop con gli agenti Cowork come modo principale di lavorare. Gli agenti sono piccoli assistenti specializzati, ciascuno con la sua istruzione che descrive cosa fa e come. Uno sviluppatore può avere agenti per analizzare codice, eseguire test, aggiornare documentazione. Un consulente può avere agenti specializzati su documenti dei clienti. Un analista può avere agenti che producono report periodici dai file locali.

Le istruzioni Cowork contengono la maggior parte del lavoro di configurazione. Ogni agente ha la sua istruzione, che definisce ruolo, file di lavoro, modalità di esecuzione, vincoli operativi. Gli agenti possono essere molti, ciascuno con istruzioni specifiche, e tipicamente fanno riferimento a file locali sul disco.

Le istruzioni di progetto possono comparire in coppia con quelle Cowork, quando un cliente specifico ha un proprio progetto di lavoro. In quel caso le istruzioni di progetto definiscono il contesto del cliente, e quelle Cowork specializzano gli agenti per quel lavoro. Quando invece gli agenti sono trasversali a più clienti, le istruzioni di progetto sono leggere o assenti.

Le istruzioni generali sono medie. Contengono preferenze trasversali come lingua e registro, eventualmente convenzioni di metodo che valgono sia per gli agenti che per le altre conversazioni. Non sono il livello principale, ma non sono nemmeno marginali come nel profilo precedente, perché chi usa gli agenti Cowork potrebbe voler lavorare anche in conversazioni libere, dove le generali contano.

Gli stili di scrittura, secondo la dinamica vista al capitolo 6, possono propagarsi anche agli output degli agenti. Restano comunque uno strumento operativo per la Chat, non sono il livello dove si concentra la configurazione di questo profilo.

La configurazione che ne risulta è centrata sugli agenti Cowork. Cowork denso, generali medie, progetti densi solo per clienti specifici, stili come strumento di Chat. È la scelta tipica di chi ha trasformato Claude in un assistente automatizzato per attività ricorrenti, dove gli agenti sostituiscono il dialogo libero su compiti specifici.

Lavoro per output singoli

Rientra in questo profilo chi usa Claude in conversazioni isolate, senza che ci sia un contesto durevole sopra. Le chat sono occasionali, ognuna nasce per un compito e si chiude lì. Vale per chi resta in Chat, per tradurre un testo, verificare un fatto, scrivere un'email difficile. Vale anche per chi su Desktop usa Cowork in modo puntuale, lanciando un agente per un compito specifico senza un'attività continuativa attorno.

Le istruzioni generali contengono la maggior parte della configurazione. Tutte le preferenze, le regole di lavoro, le modalità di risposta, eventualmente il formato di output preferito. Quando ogni chat parte da zero e si chiude in sé stessa, l'unico modo di avere una configurazione stabile è metterla nelle generali.

Le istruzioni di progetto sono assenti o usate poco. Quando il lavoro non è strutturato attorno a temi durevoli, i progetti non danno vantaggi rispetto al lavoro in Chat libera. Qualcuno crea un progetto per casi che si ripetono spesso, ma è un'eccezione.

Cowork tipicamente non rientra in questo profilo. Gli agenti sono pensati per un'attività continuativa, e chi lavora per output singoli non ha operazioni ricorrenti da affidare. L'unica eccezione è chi su Desktop usa occasionalmente un agente per un compito puntuale, come detto sopra.

Gli stili di scrittura acquistano un ruolo importante in questo profilo. Servono per modulare al volo l'output della singola chat, uno stile più conciso per le risposte rapide, uno più esplicativo per le spiegazioni, uno specifico per le email. Sono utili proprio perché la configurazione cambia frequentemente fra una chat e l'altra.

La configurazione che ne risulta concentra tutto su generali e stili. Generali dense, progetti assenti o rari, Cowork tipicamente non usato, stili come strumento operativo. È la scelta tipica di chi usa Claude come strumento di lavoro puntuale, senza una struttura organizzativa attorno.

In pratica, molti utenti non rientrano puramente in uno solo dei tre profili. Possono avere alcuni progetti densi per certi clienti, e usare Claude in conversazioni libere per il resto del lavoro. Possono avere uno o due agenti Cowork per attività ricorrenti, e poi lavorare in Chat per tutto il resto. La configurazione di chi sta in mezzo combina elementi dei tre profili, in proporzioni che dipendono dal lavoro che si fa. La cosa importante è capire come si lavora di solito, e di conseguenza dove serve mettere le regole.

Promozione e retrocessione delle regole

La configurazione non si fa una volta e basta. Cambia nel tempo, perché cambia il modo di lavorare e perché l'uso quotidiano fa scoprire cose nuove. Una regola scritta in un certo punto può rivelarsi al posto sbagliato dopo qualche settimana. Capirlo, e spostarla, fa parte della manutenzione della configurazione.

Il primo movimento possibile è la promozione. Una regola scritta nelle istruzioni di un progetto si rivela, con il tempo, utile per tutti i progetti. Magari riguarda un modo di formattare l'output, una preferenza sulle citazioni, una convenzione sui file da produrre, un certo modo di chiudere le risposte. Quando ci si accorge di ripeterla in più progetti, c'è il segnale che andrebbe spostata in alto, nelle generali, dove vale automaticamente per tutto.

Va considerata anche l'azione contraria, la retrocessione. Una regola scritta nelle istruzioni generali si rivela, con il tempo, valida solo per certi tipi di lavoro. Magari una convenzione sul formato delle date vale solo per i progetti editoriali, non per quelli di formazione aziendale. Magari una preferenza sul registro vale solo per i lavori con clienti esterni, non per quelli interni. In questi casi conviene scendere al livello giusto, perché tenere la regola nelle generali la applica anche dove non è opportuno.

Riconoscere questi spostamenti richiede di rileggere ogni tanto la propria configurazione, soprattutto dopo qualche mese di uso. I segnali da cercare sono concreti. Una regola che si trova ripetuta in più progetti è una candidata alla promozione. Una regola nelle generali che richiede continui distinguo nelle istruzioni di progetto («tranne che per il progetto X», «escluso il caso Y») è una candidata alla retrocessione.

Tenere viva questa attenzione vuol dire avere una configurazione che si adatta al lavoro effettivo. Non solo a quello che si pensava di fare quando si è scritta inizialmente.

Convivenza e ridondanza

Una stessa regola può trovarsi scritta in più posti della configurazione. A volte è una svista, altre volte una scelta deliberata. Distinguere i due casi è importante, perché tenere la stessa regola in più posti senza un motivo crea problemi quando arriva il momento di modificarla.

Il principio operativo è uno solo. Si scrive una regola in un certo livello solo quando dice qualcosa di diverso da quanto vale già a un livello precedente. Se le generali fissano l'italiano e il progetto è in italiano, non serve riscriverlo. La regola delle generali si applica automaticamente. Se il progetto è in inglese, allora va scritto.

Non tutte le apparenti ripetizioni sono ridondanze, come visto trattando il meccanismo del rafforzamento nella prima sezione, pure una regola di progetto che dice qualcosa di diverso dalle generali è un'alternativa. Sono casi in cui due regole esistono in due livelli perché ciascuna porta un contributo distinto.

La ridondanza tossica è quando la stessa regola, identica, è scritta in due posti senza che il secondo aggiunga nulla. Capita per «sicurezza», o perché si è dimenticato di aver scritto la regola da un'altra parte. Oppure è presente in due livelli diversi senza correlazioni, capita quando si scrive a distanza di tempo senza verificare cosa c'è già. In queste situazioni il problema non è immediato, si presenta quando arriva il momento di modificare la regola, perché va aggiornata ovunque sia scritta, altrimenti la configurazione resta incoerente.

Quando una ridondanza non necessaria emerge, vale la pena ripulirla. La regola va lasciata nel livello dove serve davvero, e tolta dagli altri. Se vale per tutti i lavori si lascia nelle generali. Se vale solo per un progetto si lascia nelle istruzioni di progetto. Tenerla in un solo posto rende la configurazione più chiara e più facile da mantenere.

Scenari di configurazione completa

La sezione precedente ha descritto i profili di configurazione in termini generali. Qui si vedono due scenari completi, due esempi concreti di come la configurazione si presenta per un lavoro reale. Servono come ancora per chi sta costruendo o rivedendo la propria, e mostrano come i meccanismi visti nelle sezioni precedenti si combinano nella pratica.

Consulente di formazione con più clienti

Il primo scenario è quello di un consulente che si occupa di formazione aziendale e lavora con più clienti in parallelo. Tre o quattro aziende attive, ciascuna con il suo settore, i suoi prodotti, il suo linguaggio interno. Il profilo è quello del lavoro per progetti, descritto nella sezione precedente. La maggior parte della configurazione sta nei progetti, le generali sono leggere.

Nelle generali ci sono poche regole, quelle che valgono per qualunque cliente. La lingua italiana, il registro divulgativo professionale, la preferenza per output strutturati con paragrafi narrativi più che con liste puntate, una nota sul fatto di evitare frasi enfatiche o di tipo pubblicitario.

Ogni cliente ha le sue istruzioni di progetto, dense e specifiche. Per un cliente del settore industriale c'è una descrizione dell'azienda, della linea di prodotto principale, dei termini tecnici da preferire, delle modalità di comunicazione interne adottate. Per un cliente del settore finanziario c'è una descrizione del posizionamento, della terminologia di compliance, delle convenzioni per i documenti tecnici. Le istruzioni di progetto contengono la maggior parte della configurazione di questo profilo, possono arrivare a varie pagine ciascuna.

Cowork non rientra in questo scenario. Il consulente lavora interamente in Chat, dove i progetti gli danno il contesto necessario per ogni cliente.

Gli stili di scrittura sono usati a volte per aggiustamenti puntuali. Uno stile per le email ai clienti, più sintetico e formale. Uno stile per i documenti formativi, più articolato. Niente stili che riguardino il contenuto, perché il contenuto è già nei progetti.

La configurazione complessiva è asimmetrica. Generali leggere e quasi invariabili, progetti densi e diversi tra loro, stili come strumento di adattamento dell'output al singolo compito solo quando necessario. Il consulente sa che ogni nuovo cliente comporta un nuovo progetto da configurare, ma sa anche che le generali resteranno stabili.

Sviluppatore con agenti Cowork

Il secondo scenario è quello di uno sviluppatore freelance che lavora con Claude Desktop e usa gli agenti Cowork per automatizzare i compiti ricorrenti del proprio lavoro. Code review, generazione di test, produzione di documentazione, refactoring. Il profilo è quello del lavoro con agenti Cowork. La maggior parte della configurazione sta nelle istruzioni Cowork, una per agente.

Nelle generali ci sono preferenze trasversali. La lingua italiana per le risposte conversazionali, lo stile sintetico nelle spiegazioni tecniche, una lista delle tecnologie preferite (linguaggio di programmazione principale, framework usati), una nota sul fatto di motivare brevemente le scelte tecniche proposte. Le generali servono anche per le chat libere che lo sviluppatore apre fuori dagli agenti, per chiedere consigli rapidi o spiegazioni.

Le istruzioni di progetto sono presenti per il cliente continuativo con cui lavora da tempo, e contengono il contesto del prodotto su cui lavora. Per altri clienti più occasionali non ha progetti dedicati. Cowork e progetti si combinano per le attività legate al cliente continuativo, dove gli agenti operano sui suoi file.

Le istruzioni Cowork sono il livello principale. Lo sviluppatore ha configurato quattro agenti specializzati. Un agente per la code review, che esamina pezzi di codice e segnala problemi. Un agente per la generazione di test, che produce test unitari da specifiche. Un agente per la documentazione, che produce docstring e file README da codice esistente. Un agente per il refactoring, che propone migliorie strutturali. Ogni agente ha una propria istruzione dettagliata su cosa fare, quali file di lavoro toccare, quali vincoli rispettare.

Gli stili di scrittura sono usati raramente. Uno stile «sintetico» per quando lo sviluppatore lavora a problemi rapidi nella Chat, dove vuole risposte stringate e codice essenziale. La propagazione agli agenti Cowork, vista al capitolo 6, non è importante in questo scenario. Gli agenti hanno già nelle loro istruzioni Cowork specifiche regole su come scrivere gli output.

La configurazione complessiva è centrata sugli agenti. Generali medie e stabili, un progetto attivo per il cliente continuativo, Cowork denso con più agenti, stili come strumento puntuale. Lo sviluppatore aggiunge agenti quando si presentano nuove esigenze ricorrenti, e modifica le istruzioni dei singoli agenti quando cambiano i requisiti del lavoro.

I due scenari mostrano configurazioni diverse, ognuna sensata per il proprio modo di lavorare. Non c'è una configurazione giusta in assoluto, c'è quella che corrisponde al lavoro che si fa. Costruire la propria configurazione vuol dire decidere quali livelli usare, come distribuire le regole, come gestire il cambiamento nel tempo. L'uso quotidiano fa il resto, mostrando dove serve aggiustare e dove la configurazione tiene.

Altre fonti di configurazione

I capitoli precedenti hanno descritto i livelli che l'utente configura direttamente, dalle istruzioni generali agli stili di scrittura. Sono le fonti che agiscono per scelta esplicita, ed è su queste che il manuale si è concentrato. Esistono però altri elementi che concorrono al comportamento di Claude in una conversazione, e che non rientrano in nessuno dei quattro livelli. Alcuni sono attivati indirettamente dall'utente, come le skill e i server MCP. Altri sono fuori dal suo controllo, come i blocchi di testo che Anthropic inserisce nel prompt di sistema in determinate condizioni. Altri ancora vengono ricavati automaticamente dall'attività precedente, come la memoria e le chat passate.

Conoscere queste fonti aiuta a interpretare i casi in cui Claude si comporta in modi che la sola lettura della propria configurazione non spiega. Quando il tono di una chat lunga cambia improvvisamente, quando un comportamento atteso in un progetto non si verifica perché un'informazione viene da una conversazione di mesi prima, quando l'attivazione di una skill modifica il modo in cui Claude affronta una richiesta, la causa può stare in una di queste fonti.

Il capitolo le presenta una per una, dalle più trasparenti alle meno documentate, e chiude con un'indicazione di metodo su come riconoscerle e tenerne conto.

Le skill come fonte di configurazione

Le skill sono pacchetti di istruzioni e risorse che Claude carica selettivamente quando riconosce di averne bisogno per un compito specifico. Una skill di assistenza alla scrittura contiene ad esempio le regole stilistiche e le convenzioni redazionali da applicare quando si produce un testo per una destinazione editoriale precisa. Una skill di creazione di presentazioni contiene le indicazioni su come strutturare le slide, applicare un template, esportare il file. Sono moduli pensati per coprire un dominio o un tipo di lavoro, e si aggiungono al contesto della conversazione solo quando la richiesta lo richiede.

A differenza dei livelli configurabili, che entrano sempre nel prompt di sistema all'avvio della chat, le skill funzionano in modo diverso. Ciò che entra sempre nel prompt è il loro nome e la loro descrizione, in modo che Claude sappia che la skill esiste e cosa fa. Il contenuto effettivo, cioè le istruzioni vere e proprie, viene caricato solo quando Claude valuta la skill pertinente alla richiesta in corso. È un meccanismo a due tempi, che permette di tenere a disposizione molte skill senza saturare il contesto con il testo di tutte.

Le skill richiedono che sia attiva l'esecuzione del codice e la creazione di file (*Code execution and file creation* nelle impostazioni di Claude). Senza quella funzione attivata, le skill di Anthropic non possono produrre i loro output, e l'intera sezione delle skill nelle impostazioni resta inaccessibile. È il prerequisito di base per qualsiasi uso delle skill.

Vanno distinte tre tipologie, a seconda di dove vivono le skill.

- **Le skill di Anthropic**. Sono pre-installate per tutti gli utenti e coprono compiti comuni e ricorrenti, ad esempio la creazione di documenti Word, presentazioni PowerPoint, fogli Excel e PDF. Claude le invoca automaticamente quando la richiesta lo richiede, senza bisogno di richiamarle esplicitamente.
- **Le skill personali**. Si possono creare proprie skill seguendo le convenzioni descritte da Anthropic, comprimerle in un file ZIP e caricarle in *Customize > Skills* dall'interfaccia di claude.ai. Una volta caricata, la skill è disponibile in tutte le conversazioni dell'account, sia su web sia in Claude Desktop, finché resta attiva. La skill personale è il modo per estendere stabilmente il comportamento di Claude su un dominio specifico, ad esempio le convenzioni redazionali di una testata, lo stile di una collana editoriale, le regole di un'attività ricorrente.
- **Le skill locali**. Sono skill personali che vivono nel filesystem del proprio computer, in una posizione specifica del profilo di Claude Desktop o di Claude Code. Funzionano solo dentro questi ambienti e solo sul computer in cui sono installate. Non risiedono sui server di Anthropic, e non sono visibili dagli altri device collegati all'account. Sono adatte ai casi in cui la skill contiene riferimenti a file locali, percorsi del filesystem, strumenti installati sul computer. Oppure quando si vuole abilitare il loro uso su un singolo computer.

La distinzione fra le tre tipologie ricalca, sul piano dell'estensione di Claude, la stessa logica già vista al capitolo 4 per le istruzioni di progetto. Una skill di Anthropic è portabile per costruzione, perché è disponibile ovunque, come lo è una skill personale portabile dentro l'account, perché Anthropic la rende accessibile sia su web sia in Desktop. Una skill locale non è portabile, perché vive in un singolo ambiente. Le considerazioni viste nella sezione *La stessa istruzione, ambienti diversi* del capitolo 4 valgono qui con la stessa logica.

L'effetto delle skill sul prompt di sistema effettivo si compone con quanto già detto al capitolo 2. Quando una skill viene caricata, il suo contenuto si aggiunge al contesto della conversazione come istruzioni aggiuntive, che convivono con le istruzioni generali, di progetto, di Cowork e con lo stile attivo. Le regole di prevalenza continuano a valere. Una skill che impone un certo formato per le presentazioni è specifica per quel tipo di output, e prevale rispetto a una regola generale sul formato delle risposte. Una skill che parla di un dominio diverso da quello del progetto non confligge, si combina semplicemente.

```
La gestione delle skill avviene nella sezione *Customize > Skills* dell'interfaccia di
Claude. Da lì si vedono tutte le skill disponibili sull'account, si attiva o si disattiva
ciascuna con uno switch, e si può aprire la scheda di una skill per leggere descrizione e
contenuto. Vale la pena farlo almeno una volta per ciascuna skill che si usa, comprese
quelle di Anthropic, per capire cosa fa effettivamente. Una skill spenta non viene
caricata da Claude in nessun caso, anche quando la richiesta sarebbe pertinente. Una
skill accesa è disponibile, ma non viene necessariamente usata, perché Claude valuta caso
per caso se è pertinente alla richiesta in corso. È disponibilità, non attivazione
forzata. La descrizione che si scrive nella metadata della skill è il punto che Claude
usa per la verifica della pertinenza, una descrizione corretta permette al sistema di
gestire correttamente il caricamento corretto per una certa necessità.
```

I server MCP e le loro descrizioni

I server MCP, dall'acronimo Model Context Protocol, sono un meccanismo di estensione di Claude che permette di mettere a disposizione strumenti esterni, dalla lettura di un calendario alla gestione di file su un servizio remoto, dall'interrogazione di un database al controllo di applicazioni dedicate. Un server MCP espone uno o più tool, ciascuno con un nome e una descrizione, e Claude può invocarli durante la conversazione quando la richiesta dell'utente li rende pertinenti. Sono uno strumento operativo, non una sorgente di istruzioni esplicite, ma concorrono al comportamento di Claude e meritano una nota dedicata.

```
Il punto di intersezione con la configurazione è la descrizione dei tool. Quando si
attiva un server MCP, le descrizioni dei tool che espone entrano nel prompt di sistema
della conversazione. Servono a Claude per capire cosa fa ogni tool e quando vale la pena
invocarlo. Sono testi scritti dagli autori del server, non dall'utente, e l'utente non
può modificarli direttamente se il server non è locale. Hanno però effetti sul
comportamento di Claude paragonabili a quelli di un'istruzione, perché orientano la
scelta degli strumenti e contribuiscono al modo in cui Claude affronta certi tipi di
richiesta.
```

Quanto descritto ha un paio di conseguenze, per quanto riguarda il discorso sviluppato nel manuale. Attivare molti server MCP riempie il prompt di sistema di descrizioni, che competono fra loro per l'attenzione di Claude. Quando i tool disponibili sono molti, e in particolare quando le loro descrizioni si somigliano, può capitare che Claude scelga un tool quando ne avrebbe servito un altro, o ne invochi uno quando non era richiesto. Tenere attivi solo i server effettivamente utili al proprio lavoro è una pratica di igiene, simile a quella già vista per le skill. Inoltre la qualità di un server MCP dipende anche dalla cura delle sue descrizioni. Un server con tool ben descritti viene usato in modo più appropriato di un server con descrizioni vaghe o ambigue. Quando si sviluppa un proprio server MCP, dedicare attenzione alle descrizioni dei tool è altrettanto importante che progettarne la logica.

Come per le skill, la disponibilità dei server MCP non coincide con il loro uso. Un server attivo è disponibile, ma Claude lo invoca solo se la richiesta lo rende pertinente. Un server spento non viene proposto, anche se la sua descrizione coprirebbe la richiesta. La gestione si fa dalle impostazioni di Claude Desktop, dove i server MCP locali vengono dichiarati, e dall'interfaccia web per i server di tipo remoto.

Le iniezioni di sistema

Esiste una seconda categoria di fonti che concorrono al comportamento di Claude, e che si distingue dalle skill e dai server MCP perché non sono attivate dall'utente, neppure indirettamente. Si tratta di blocchi di testo brevi, chiamati *reminder*, che Anthropic aggiunge al prompt di sistema in autonomia, quando determinate condizioni si verificano nella conversazione. Non sono pensate per estendere le capacità di Claude su un dominio specifico, ma per richiamare regole di comportamento che il modello deve tenere presenti in situazioni che Anthropic ritiene meritino una nota in più.

Vengono inseriti dinamicamente nel prompt quando viene identificata una condizione, oppure al raggiungimento di una soglia interna come una certa lunghezza di conversazione. L'utente non li vede, ed è

progettuale che resti così, perché sono indicazioni rivolte a Claude e non al lettore.

Anthropic dichiara esplicitamente l'esistenza del meccanismo e i nomi dei reminder utilizzati. Al momento della scrittura di questo manuale sono sei: *image_reminder*, *cyber_warning*, *system_warning*, *ethics_reminder*, *ip_reminder*, *long_conversation_reminder*.

```
Non sono documentati esplicitamente i criteri di attivazione di ciascuno, né il contenuto
testuale. Quello che si può dire è che, nel loro insieme, possono modificare il modo in
cui Claude affronta una conversazione in corso. Quando una risposta sembra discostarsi
dal comportamento abituale senza una causa identificabile nella propria configurazione,
l'attivazione di un reminder è una delle ipotesi da considerare. Riconoscerlo evita di
attribuire alla propria configurazione un comportamento che dipende da una fonte esterna
su cui non si ha controllo.
```

Memoria e chat passate

Le due fonti che restano da considerare sono diverse dalle precedenti per natura. Non sono pacchetti di istruzioni come le skill, e non sono testi che Anthropic inserisce nel prompt come le iniezioni di sistema. Sono informazioni che Claude ricava dall'attività precedente dell'utente, e che entrano a comporre il contesto delle conversazioni successive. Influiscono sulla configurazione in modo indiretto, perché non sono regole formali. Incidono sul comportamento, perché ciò che Claude sa dell'utente e dei suoi lavori precedenti orienta il modo in cui legge le richieste in corso.

- **Memoria**. Quando attiva, è un meccanismo che genera in modo automatico sintesi delle chat passate dell'utente, e le conserva in un'area dedicata dell'account. Le memorie raccolgono fatti rilevanti come preferenze ricorrenti, contesti di lavoro, persone e progetti citati. Vengono inserite nel prompt di sistema delle conversazioni successive in forma sintetica. Esistono memorie di account, comuni a tutte le chat fuori dai progetti, e memorie di progetto, separate per ciascun progetto in cui sono attivate. Si vedono dalle impostazioni di Claude e si possono modificare, integrare, cancellare. L'effetto delle memorie sul comportamento è quello di una regola informale che opera in background. Se in passato si è detto a Claude di preferire risposte brevi, e la memoria ha registrato questa preferenza, Claude continuerà a dare risposte brevi nelle conversazioni successive senza che si debba ripetere la richiesta. Vale lo stesso per le abitudini di lavoro, gli interlocutori abituali, le scelte stilistiche. È simile a quanto visto al capitolo 4 per la deriva contestuale. La differenza è che qui la fonte non è una conversazione in corso, ma una storia accumulata di conversazioni precedenti.
- **Ricerca su chat passate**. Quando attiva, permette a Claude di cercare nelle chat precedenti dell'utente quando una richiesta lo suggerisce. La differenza con la memoria è che questa fonte non è preinserita nel prompt di sistema. Viene attivata su richiesta, come uno strumento di ricerca che Claude può usare quando il contesto lo richiede. Tipicamente succede quando l'utente fa riferimento a una conversazione precedente, con formulazioni del tipo «*come ti dicevo l'altro giorno…*» o «*riprendiamo quella discussione su…*». Claude cerca la chat pertinente, ne estrae i contenuti utili, e li usa come contesto per la risposta corrente. Anche per la ricerca su chat passate l'effetto sulla configurazione effettiva è indiretto ma significativo. Una decisione presa mesi prima e archiviata in una chat ora chiusa può tornare a influenzare il lavoro corrente. È un'opportunità, perché permette di riprendere il filo senza dover

rispiegare tutto. È anche una possibile sorpresa, perché un'indicazione vecchia di cui ci si è dimenticati può rientrare nel contesto e influenzare la risposta in modi non immediatamente prevedibili.

Sia la memoria sia la ricerca su chat passate sono attivabili o disattivabili in *Impostazioni -> Funzionalità*. La scelta dipende dal tipo di lavoro. Per attività continuative, dove l'accumulo di contesto fra una sessione e l'altra è un beneficio, conviene tenerle attive. Per lavori puntuali, dove la chat è autoconclusiva, può convenire tenerle spente, in modo che ogni conversazione parta dalla sola configurazione esplicita. La distinzione fra attivazione e disattivazione non è binaria. La gestione delle singole memorie permette di mantenere il meccanismo attivo intervenendo selettivamente su cosa Claude deve ricordare e cosa no.

Come riconoscere e gestire queste fonti

Le quattro famiglie di fonti descritte fin qui hanno gradi diversi di trasparenza e di controllo da parte dell'utente. La differenza è importante, perché determina cosa si può fare e cosa si può solo riconoscere.

Le skill e i server MCP sono i casi più trasparenti. Le skill sono visibili in *Customize > Skills*, si possono leggere, attivare e disattivare singolarmente. I server MCP si gestiscono dalle impostazioni di Claude Desktop o dell'interfaccia web a seconda del tipo, e per ciascuno l'utente può scegliere se tenerlo attivo o no. Chi ne sviluppa di propri ha pieno controllo sul contenuto e sulla descrizione. La gestione consiste nel mantenere elenchi coerenti con i lavori che si fanno, disattivando ciò che non serve e tenendo acceso solo ciò che è pertinente. Una skill o un MCP che restano accesi senza essere usati non danneggiano, ma una descrizione vaga o ambigua può portare a invocazioni inattese, oppure a mancate invocazioni dove sarebbero servite.

Le iniezioni di sistema sono il caso meno trasparente. Non sono visibili dall'utente, non sono configurabili, non sono disattivabili. Il loro funzionamento è quello che Anthropic decide al momento, e può variare nel tempo. Quello che si può fare è imparare a riconoscerne gli effetti. Un cambio improvviso di tono in una chat lunga, una cautela inattesa su un tema di copyright, una richiesta che riceve una risposta più asciutta del solito sono i segnali tipici. Riconoscerli evita di attribuire alla propria configurazione un comportamento che dipende da un'iniezione esterna. Per il *long_conversation_reminder*, una tecnica usata dalla community è inserire nel campo *Profilo* delle istruzioni generali una formulazione che chieda a Claude di mantenere il tono abituale anche nelle chat lunghe. Non disattiva il reminder, ma ne può attenuare l'effetto.

Memoria e chat passate stanno nel mezzo. Sono trasparenti come le skill, perché si vedono e si modificano dalle impostazioni di Claude. Hanno un grado di indeterminatezza maggiore, perché il loro contenuto cresce nel tempo con l'uso. Ciò che a un certo punto orienta il comportamento di Claude può non essere quello che si pensava. Una revisione periodica delle memorie, fatta a freddo, è il modo più diretto per intercettare informazioni vecchie o errate prima che producano effetti. La stessa logica vale per la ricerca su chat passate. La scelta concreta è se tenerla attiva, sapendo che ogni conversazione precedente può rientrare nel contesto, oppure disattivarla, accettando di dover ricostruire il contesto quando serve.

Il principio di metodo che chiude il capitolo è quello già sotteso a tutto il manuale. Una configurazione di base ordinata, scritta con disciplina e mantenuta nel tempo, è il modo per gestire i comportamenti di Claude. Quando un comportamento è coerente con la propria configurazione, viene da lì. Quando non lo è, conviene cercare la causa in una delle fonti descritte in questo capitolo, prima di intervenire sulla configurazione stessa. Modificare le istruzioni per correggere un comportamento che in realtà dipende da una skill invocata fuori contesto, da un tool MCP che ha preso il posto sbagliato, da un'iniezione di sistema o da una memoria datata, vuol dire aggiungere rumore alla configurazione senza affrontare la causa effettiva. La pulizia della propria configurazione è anche uno strumento diagnostico.

La forza delle regole

Il capitolo 2 ha introdotto due dimensioni della prevalenza fra livelli di configurazione, la gerarchia e la forza, ed ha esposto la prima in dettaglio. La prevalenza gerarchica orienta correttamente la maggior parte delle scelte di configurazione, perché copre i casi in cui le regole in conflitto sono formulate con intensità di pari livello. Per questo motivo i capitoli successivi hanno lavorato a parità di forza, assumendo che le regole fossero scritte con la stessa intensità e che la prevalenza fosse decisa dalla sola posizione del livello.

```
Esiste però una seconda dimensione, anticipata dal capitolo 2 e già incontrata con il
caso della lingua nel capitolo 4. Quando le istruzioni in conflitto sono formulate con
intensità diverse, la più forte può prevalere anche se sta in un livello meno specifico.
La forza di un'istruzione, intesa come intensità della formulazione e capacità di
resistere nel contesto della conversazione, è una variabile autonoma rispetto alla
gerarchia. Nei casi in cui le due dimensioni convergono, cioè quando la regola scritta
nel livello più specifico è anche la più forte, il risultato è quello che la sola regola
gerarchica avrebbe previsto. Nel caso opposto il risultato può essere diverso.
```

Il capitolo descrive i fattori che determinano la forza di un'istruzione, le situazioni in cui la forza ribalta la prevalenza gerarchica, e le indicazioni pratiche per scrivere regole con la forza adeguata al loro scopo. I fattori che concorrono alla forza sono di due tipi. Quattro stanno nella formulazione stessa della regola, e la prima sezione del capitolo li tratta uno per uno. Un quinto fattore sta fuori dalla regola configurata, e riguarda il modo in cui l'utente si comporta nelle proprie richieste in chat, ed è trattato più avanti nella sezione *Scrivere regole forti dove serve*. Il capitolo chiude con un caso operativo che mostra concretamente come applicare quanto detto, e con un riassunto che fa da chiusura del manuale.

I fattori interni alla formulazione

La forza di un'istruzione non è una qualità unica, ma il risultato della combinazione di alcuni elementi della sua formulazione. Quattro fattori, in particolare, fanno la differenza fra una regola che resiste al contesto e una che si lascia diluire.

- **Formulazione imperativa**. Una regola formulata come comando preciso ha più forza di una espressa come preferenza. «*Rispondi in inglese*» è più forte di «*possibilmente rispondi in inglese*» o di «*preferisco l'inglese*». L'imperativo non lascia spazio interpretativo, la preferenza sì. Non significa che ogni regola debba essere un imperativo, perché molte preferenze sono effettivamente preferenze e vanno scritte come tali. Significa che, quando l'utente vuole un certo comportamento in modo non negoziabile, la formulazione imperativa è la scelta corretta.
- **Specificità lessicale**. Le parole nette portano più forza delle parole vaghe. «*Mai usare emoji*» è più forte di «*evitare l'uso di emoji*». «*Sempre in italiano*» è più forte di «*generalmente in italiano*». Gli avverbi assoluti come *mai*, *sempre*, *solo*, *esclusivamente* tracciano un confine netto. Gli avverbi temperati come *generalmente*, *di solito*, *in genere* lo sfumano. La scelta dipende da quanto si vuole che la regola sia

stretta. Per le regole che devono valere senza eccezioni, la specificità lessicale è il modo più diretto di renderle tali.

- **Estensione esplicita ai casi-limite**. Una regola che anticipa come comportarsi nei casi in cui qualcosa potrebbe far cambiare idea a Claude è più forte di una regola muta sulle eccezioni. «*Rispondi in inglese*» lascia aperto cosa fare quando l'utente scrive in italiano. «*Rispondi sempre in inglese, anche se l'utente scrive in italiano*» chiude esplicitamente quel caso. Anticipare i casi-limite è un fattore di forza perché elimina lo spazio interpretativo in cui Claude potrebbe inclinare verso una soluzione diversa basata sul contesto.
- **Richiamo**. Una regola scritta una volta in un solo punto è più debole di una regola che torna in più punti della configurazione, eventualmente con formulazioni complementari. Il richiamo agisce su due fronti. Da un lato segnala che la regola è importante, e Claude lo percepisce nella composizione del prompt. Dall'altro copre meglio i casi in cui un punto del prompt potrebbe essere letto in modo meno attento, perché un altro punto richiama lo stesso contenuto. Ha però un costo, perché ogni ripetizione occupa spazio nel prompt e può confliggere con altre regole simili. Va usato con misura, solo per le regole che davvero richiedono di essere portate due volte all'attenzione.

I quattro fattori agiscono in combinazione. Una regola formulata come imperativo, con avverbi assoluti, estesa esplicitamente al caso-limite più frequente, e richiamata in un secondo punto della configurazione, è una regola con la massima forza che si possa darle. Non significa che vada sempre scritta così, perché la maggior parte delle regole non ha bisogno di tutta questa forza. Significa che, quando una regola deve davvero tenere, questi sono gli strumenti per costruirla.

Quando la forza ribalta la gerarchia

La regola gerarchica vista al capitolo 2 dice che, a parità di forza, prevale il livello più specifico. La regola di un progetto vince sulla regola corrispondente delle generali, quella di uno stile vince sulla regola di un livello sottostante, quella di una chat vince su tutte. Questa regola copre la maggior parte dei casi reali, perché in genere chi configura il proprio sistema scrive con uno stile coerente fra livelli, e le differenze di forza fra una regola e l'altra sono piccole o nulle.

Quando però le forze sono significativamente diverse, la regola gerarchica può non bastare a prevedere il comportamento. Tre scenari rendono chiaro il punto.

- **Convergenza**. Forza e gerarchia vanno nella stessa direzione, perché la regola scritta nel livello più specifico è anche quella formulata con più intensità. Il vincitore è chiaro, e non c'è ambiguità da risolvere. È lo scenario più frequente quando l'utente cura la propria configurazione, perché la regola specifica viene scritta apposta per un contesto, e per quel contesto la si scrive con la dovuta nettezza.
- **Divergenza marginale**. La regola gerarchicamente più specifica è formulata con un po' meno forza di quella corrispondente nel livello superiore. Per esempio, nelle generali c'è una regola netta, nel progetto c'è una formulazione un po' più morbida sullo stesso punto. La prevalenza gerarchica regge. La regola di progetto, anche se formulata in modo meno enfatico, vince comunque, perché la differenza di forza

non è sufficiente a ribaltare la specificità del livello. Le sfumature di formulazione contano, ma fino a un certo punto.

- **Ribaltamento**. La regola gerarchicamente meno specifica è formulata con forza notevolmente maggiore di quella corrispondente nel livello più specifico. Per esempio, nelle generali c'è una regola scritta come imperativo, con avverbi assoluti, estesa esplicitamente ai casi-limite, e richiamata in più punti. Nel progetto c'è una regola morbida che prova a discostarsene, oppure è espressa in modo minimale. La regola delle generali può prevalere anche se è gerarchicamente meno specifica, perché la differenza di forza compensa e supera la differenza di specificità. È il caso in cui la dimensione della forza ribalta la prevalenza gerarchica.

Va detto che il ribaltamento è una tendenza, non un meccanismo deterministico. Claude non applica una regola di calcolo per stabilire quale fattore vinca. La prevalenza è una proprietà della composizione del prompt di sistema, e la composizione del prompt include sia la specificità del livello sia l'intensità della formulazione. In presenza di differenze marcate di forza, la più forte tende a prevalere. In presenza di differenze marginali, la più specifica continua a vincere. Fra i due estremi c'è una zona di incertezza, in cui il comportamento può essere meno prevedibile.

L'implicazione pratica è una sola. Quando si vuole che una regola specifica prevalga effettivamente su una corrispondente regola del livello superiore, la specificità deve essere accompagnata da una forza almeno comparabile. Una regola morbida nel livello specifico, opposta a una regola netta nel livello generale, è uno squilibrio che, dal punto di vista del comportamento osservato, può non avere l'esito atteso.

Scrivere regole forti dove serve

Tutto quanto detto fin qui suggerisce una pratica disciplinata, non un uso indiscriminato. Scrivere regole forti ovunque non rende la configurazione più efficace, la rende più pesante. Una configurazione in cui ogni regola è formulata con imperativi, avverbi assoluti ed estensioni esplicite finisce per appesantire il prompt di sistema con segnali che competono fra loro. Tutto sembra urgente, e in effetti nulla lo è.

La pratica utile è opposta. Il default va tenuto leggero, con formulazioni piane che esprimono la regola nella sua forma essenziale. La forza si aggiunge selettivamente, dove serve, e per ragioni precise.

Tre indicazioni operative aiutano a capire dove serve.

- **Le regole su elementi sensibili al contesto**. Lingua, tono, registro, formato di risposta, lunghezza. Sono gli aspetti su cui Claude tende a bilanciare l'istruzione configurata con i segnali della conversazione. Per queste regole la forza serve quasi sempre, perché il contesto della chat può spingere in direzioni diverse da quanto previsto. La formulazione netta con estensione esplicita ai casi-limite è la scelta corretta.
- **Le regole non negoziabili per il proprio lavoro**. Convenzioni redazionali di una testata, regole metodologiche di un progetto critico, divieti specifici legati a un cliente. Sono regole che, se cedono, producono un output non utilizzabile. Per queste regole la forza è un investimento ragionevole, perché il costo di un fallimento occasionale è alto.

- **Le regole che si è già visto Claude faticare a tenere**. La pratica nel tempo mostra dove le formulazioni piane non bastano. Quando si nota che un certo comportamento non viene tenuto in modo affidabile, il primo intervento è rinforzare la formulazione, non aggiungere nuove regole. Una regola scritta una volta sola, in modo morbido, in un livello qualunque, è spesso la causa del problema. La soluzione è riformularla nei termini visti nella prima sezione di questo capitolo.

Per tutte le altre regole, la formulazione semplice basta. Una preferenza espressa come preferenza, un suggerimento di stile come suggerimento, una linea di metodo come linea di metodo. Sono regole che vivono bene anche senza enfasi, perché Claude le legge nel prompt insieme alle altre, e le applica quando il contesto le richiede.

C'è infine un fattore di rinforzo che non sta nella formulazione della regola, ma nel comportamento dell'utente. È stato anticipato al capitolo 4 nella sezione *Le regole di prevalenza alla prova del contesto*, e vale la pena richiamarlo qui come componente della forza. Una regola che richiede a Claude un certo comportamento risulta più forte quando l'utente, nelle sue richieste in chat, agisce come se la regola fosse già operativa. Se la regola di un progetto richiede l'inglese, scrivere in inglese nelle proprie richieste a Claude rinforza la regola, perché allinea il contesto della conversazione con la regola configurata. L'incoerenza fra istruzione e comportamento dell'utente è essa stessa un segnale che indebolisce la regola, indipendentemente da come è formulata.

Un caso operativo verificato

Il capitolo 4 ha presentato il caso della lingua da usare. La configurazione era italiano nelle generali, inglese nel progetto, utente che scrive in italiano nelle chat di quel progetto.

La regola di progetto da cui si partiva era, nella sua forma minima, «*rispondi in inglese*». L'imperativo c'è, la lingua è indicata, ma il caso-limite del messaggio in italiano resta aperto. Per Claude, fra un'istruzione del prompt di sistema che richiede l'inglese e un messaggio dell'utente scritto in italiano, lo spazio interpretativo basta a far prevalere il contesto.

La riformulazione opera su due dei quattro fattori visti nella prima sezione. La formulazione imperativa era già presente nella regola originale, e il richiamo non si applica al caso, perché la regola sta in un solo punto della configurazione e non c'è ragione di duplicarla altrove. I due fattori da aggiungere sono la specificità lessicale e l'estensione esplicita al caso-limite.

- **Specificità lessicale**. All'imperativo si aggiunge un avverbio assoluto. «*Rispondi sempre in inglese*» è più forte di «*rispondi in inglese*». Il *sempre* segnala che la regola non ha eccezioni di contesto.
- **Estensione esplicita al caso-limite**. Va dichiarata la condizione in cui Claude tenderebbe a discostarsi, e va detto cosa fare in quella condizione. «*Rispondi sempre in inglese, anche se l'utente scrive in italiano*» copre esplicitamente il caso che produce la deriva. Lo spazio interpretativo si chiude.

A questi fattori si aggiunge, secondo quanto visto nella terza sezione, la coerenza del comportamento dell'utente. Scrivere in inglese nelle proprie richieste a Claude dentro le chat di quel progetto allinea il

contesto della conversazione con la regola configurata. Anche una regola formulata bene fatica a tenere quando l'utente, in chat, si comporta come se la regola non valesse.

Applicati insieme, questi elementi riducono significativamente la probabilità che il comportamento di Claude in quel progetto si discosti dall'inglese, anche quando arrivano messaggi in italiano. La probabilità non è azzerata, e questo è coerente con quanto detto nella seconda sezione. La prevalenza è una tendenza, non un calcolo deterministico. Una regola scritta con la forza necessaria vince nei casi normali, ma può cedere in situazioni particolari. Sono casi marginali, e la pratica nel tempo mostra che la riformulazione descritta tiene nella stragrande maggioranza delle conversazioni reali.

```
## In sintesi

Cinque principi riassumono quanto detto nel capitolo, e chiudono il quadro del manuale.

1. La prevalenza fra livelli ha due dimensioni. La prima è la gerarchia, descritta al
capitolo 2, che a parità di forza fa vincere il livello più specifico. La seconda è la
forza dell'istruzione, intesa come intensità della formulazione e capacità di resistere
al contesto della conversazione.

2. Quattro fattori determinano la forza di una regola, e si combinano fra loro. Sono la
formulazione imperativa, la specificità lessicale, l'estensione esplicita ai casi-limite,
il richiamo della regola in più punti della configurazione.

3. Quando le forze sono di pari livello, la gerarchia decide. Quando una regola
gerarchicamente meno specifica è formulata con forza notevolmente maggiore di una più
specifica, la prima può prevalere. È il caso in cui la dimensione della forza ribalta la
prevalenza gerarchica.

4. La pratica corretta è tenere leggera la formulazione di default e aggiungere forza
selettivamente. Le regole su elementi sensibili al contesto, quelle non negoziabili e
quelle che Claude ha già faticato a tenere meritano forza. Le altre stanno bene nella
forma piana.

5. Anche il comportamento dell'utente in chat funge da rinforzo. La coerenza fra
istruzione configurata e modo di scrivere nelle proprie richieste aumenta la forza
effettiva della regola, indipendentemente dalla sua formulazione testuale.
```

Il manuale si chiude qui, dopo aver presentato i livelli di configurazione, le loro regole di composizione, le altre fonti che concorrono al comportamento, e le due dimensioni della prevalenza che orientano la scelta delle regole. L'impressione complessiva è che la configurazione di Claude non è un'operazione di compilazione, ma una pratica di disciplina che si costruisce nel tempo. È una pratica che richiede la stessa cura che si dedica a un proprio metodo di lavoro.

Le regole scritte una volta sola e dimenticate poco dopo contano poco. Quelle scritte con attenzione, rilette periodicamente, adattate man mano che il proprio lavoro evolve, fanno la differenza. La pulizia della configurazione, la consapevolezza delle altre fonti che vi si affiancano, l'attenzione alla forza dove serve e alla leggerezza dove basta sono gli strumenti di chi vuole usare Claude in modo affidabile su lavori che richiedono continuità.