

HT2 ACADEMY: VOLUME 1

Il viaggio del dato

Come l'intelligenza artificiale trasforma l'informazione
aziendale in azione

HT2 TEAM

Prima edizione — 2026

Il viaggio del dato

Come l'intelligenza artificiale trasforma l'informazione aziendale in azione

HT2 Team

Copyright © 2026 HT2 www.ht2.it

Tutti i diritti riservati.

Prima edizione: 2026

Composto in L^AT_EX con font Palatino.

Ringraziamenti

Desidero esprimere la mia più profonda gratitudine a tutte le persone che hanno reso possibile la realizzazione di questo volume sull'Agentic AI. Un progetto di questa complessità non avrebbe mai visto la luce senza il contributo sinergico, la visione e la competenza tecnica di un team eccezionale.

ALLA VISIONE STRATEGICA E TECNICA DI HT2

Un ringraziamento speciale va a **Michele Russo**, **Asya Pesaresi**, **Luca Fuligni** e **Paola Spinelli**, membri del Comitato Strategico di HT2, per aver fornito la direzione e il supporto necessari a esplorare le frontiere degli agenti autonomi sia dal punto di vista tecnico che organizzativo.

Allo stesso modo, la solidità scientifica di questo lavoro è frutto dell'impegno di **Natalia Miccini**, e **Dmitry Mingazov**, membri del Comitato Tecnico, il cui rigore nell'analisi delle architetture AI è stato il pilastro su cui si fonda ogni capitolo.

ALLA NOSTRA COMMUNITY E AI NOSTRI PARTNER

Un riconoscimento sentito va alla Community di HT2 nella sua interezza. Siete voi, con il vostro entusiasmo e il continuo scambio di idee, a dare un senso concreto alla nostra ricerca e a spingerci a superare i limiti dell'innovazione.

Un ringraziamento fondamentale va inoltre a **DeepLearning.AI**. Siamo onorati di poter operare come ponte verso il loro straordinario ecosistema di corsi e know-how. Questa sinergia ci permette di democratizzare l'accesso a una formazione di livello mondiale e di trasferire le competenze più avanzate direttamente nel cuore dei nostri progetti.

Paolo Alessandroni
Presidente HT2 e Ambassador DeepLearning.AI

Prefazione

Questo libro nasce da una domanda che abbiamo sentito ripetere molte volte, in contesti diversi, da persone diverse.

Come faccio a capire davvero di cosa si sta parlando?

Non la domanda di chi vuole imparare a programmare. Non la domanda di chi insegue l'entusiasmo del momento o cerca la tecnologia per la tecnologia. La domanda di chi ha responsabilità concrete in un'organizzazione, si trova ogni giorno di fronte a decisioni che riguardano l'intelligenza artificiale, e si rende conto che le risorse disponibili sono quasi sempre o troppo superficiali o troppo tecniche. Che manca qualcosa nel mezzo. Che ci sono persone intelligenti, motivate, con esperienza reale di business, che si sentono straniere in conversazioni che le riguardano direttamente.

Noi di Ht2 lavoriamo da anni al confine tra tecnologia e organizzazione. Accompagniamo aziende di settori diversi — manifattura, servizi, finanza, sanità, distribuzione — nel momento in cui decidono di fare un passo serio verso l'adozione dell'AI. Non il chatbot che impressiona nelle demo e scompare in produzione dopo tre mesi. I sistemi che cambiano davvero il modo in cui il lavoro viene fatto, le decisioni vengono prese, il valore viene creato.

In quel percorso, la cosa che fa la differenza non è quasi mai la tecnologia. Non è la scelta del modello, non è l'architettura del sistema, non è la qualità del codice. Tutti elementi importanti, certo — ma non il fattore critico.

Il fattore critico è la capacità delle persone con responsabilità di business di partecipare alla conversazione in modo informato. Di fare le domande giuste. Di riconoscere quando una proposta è solida e quando è marketing ben confezionato. Di valutare un progetto non solo sulla base di una demo convincente, ma sulla base di come funzionerà in produzione, con dati reali, con utenti reali, nel tempo. Di prendere decisioni consapevoli invece di delegarle interamente a chi ha competenze tecniche — oppure, all'opposto, di bloccarle per paura di non capire abbastanza.

Abbiamo visto entrambe le situazioni. Organizzazioni che hanno approvato progetti perché qualcuno nel team tecnico era entusiasta, senza che nessuno con responsabilità di business fosse nelle condizioni di valutare davvero. E organizzazioni che hanno rimandato per anni investimenti che avrebbero potuto cambiare la loro posizione competitiva, perché il linguaggio sembrava inaccessibile e la complessità intimidiva.

Questo libro è il tentativo di colmare quel divario.

Non è un manuale tecnico. Non insegna a costruire sistemi AI né a scrivere una riga di codice. È una guida per chi deve capirli, valutarli, governarli, finanziarli. Per chi siede al tavolo in cui si decide se investire, come misurare il risultato, quando fermarsi e quando accelerare. Per chi vuole essere un interlocutore competente — non un esperto, ma qualcuno che sa dove guardare e cosa chiedere. Per chi ha la responsabilità di fare andare le cose per davvero, non solo di approvarle sulla carta.

Abbiamo scelto di organizzare il contenuto attorno a un'idea semplice: il viaggio del dato. Dall'informazione grezza che dorme in un database all'azione intelligente che produce valore concreto. È un percorso che molte organizzazioni stanno cercando di fare, spesso senza una mappa chiara, spesso con aspettative sbagliate in un senso o nell'altro. Questo libro vuole essere quella mappa.

Il percorso è articolato. Partiamo da come le aziende raccolgono e strutturano le proprie informazioni, passiamo per i sistemi che le rendono interrogabili in modo intelligente, arriviamo agli agenti che trasformano la conoscenza in azione, e infine esploriamo le architetture multi-agente che permettono di affrontare la complessità reale dei processi aziendali. Ogni capitolo aggiunge uno strato. Ogni strato si appoggia su quello precedente.

C'è una cosa che teniamo a dire con chiarezza, prima che iniziate.

Non promettiamo completezza — il campo si muove troppo velocemente per qualsiasi promessa del genere. Non promettiamo che ogni progetto AI avrà successo — dipende da troppe variabili che questo libro non può controllare. Non promettiamo che la complessità scomparirà: l'AI applicata all'impresa è genuinamente complessa, e chi vi dice il contrario sta semplificando in modo fuorviante.

Quello che promettiamo è onestà. Sui limiti dei sistemi tanto quanto sulle loro possibilità. Sui costi reali, non solo su quelli visibili al momento dell'investimento. Sulle domande che hanno risposta e su quelle che ancora non ce l'hanno. Perché le decisioni migliori si prendono con informazioni accurate, non con entusiasmo non calibrato.

E promettiamo rispetto per il vostro tempo e per la vostra intelligenza. Questi argomenti possono essere spiegati senza rendere tutto banale e senza rendere tutto incomprensibile. Abbiamo cercato il punto di equilibrio — quello in cui la complessità viene presentata senza essere nascosta, e il rigore non viene sacrificato alla semplicità.

Se dopo aver letto queste pagine vi sentite più capaci di partecipare alla conversazione sull'AI nella vostra organizzazione — di fare le domande giuste, di valutare le risposte, di distinguere il sostanziale dal superficiale, di essere protagonisti invece di spettatori — allora questo libro ha fatto il suo lavoro.

Il dato ha un viaggio da fare. E adesso, anche voi.

Milano, 2026

Comitato Strategico HT2 (www.ht2.it)

Contents

Ringraziamenti	v
Prefazione	vii
Introduzione	1
1 Prima dell'AI: il caos che tutti conoscono	7
1.1 L'azienda che annega nei suoi stessi dati	7
1.2 Dati strutturati e non strutturati: la distinzione che cambia tutto	8
1.2.1 I dati strutturati: il mondo ordinato	8
1.2.2 I dati non strutturati: il mondo reale	9
1.2.3 Il confine che nessun motore di ricerca tradizionale sa attraversare	9
1.2.4 Un problema dentro il problema: il Shadow IT	10
1.2.5 Perché questa distinzione è il punto di partenza	10
1.3 Il costo nascosto dell'informazione inaccessibile	11
1.3.1 Il tempo perso a cercare: la voce più grande e più invisibile	11
1.3.2 Le decisioni prese con dati incompleti: il costo che nessuno vede	11
1.3.3 La conoscenza che evapora: il costo più irreversibile di tutti	12
1.3.4 Il costo silenzioso della duplicazione: quando si reinventa la ruota	13
1.3.5 Il costo totale: una stima che nessuno vuole fare	13
1.4 Perché le soluzioni tradizionali non bastano	14
1.4.1 La prima lapide: il portale intranet	14
1.4.2 La seconda lapide: i sistemi di knowledge management	15
1.4.3 La terza lapide: i motori di ricerca interni	16
1.4.4 La quarta lapide: le procedure scritte e la formazione sistematica	16
1.4.5 Il pattern comune: perché tutte queste soluzioni falliscono	17
1.4.6 Il cambio di paradigma che cambia tutto	17
1.5 Il dato che vuole viaggiare	18
1.5.1 Il potenziale latente: ciò che già possedete senza saperlo	18
1.5.2 Il viaggio del dato: dalla generazione all'azione	19
1.5.3 Tre condizioni perché il viaggio avvenga	19
1.5.4 Perché adesso	20
1.5.5 Quello che avete appena riconosciuto	21
	ix

2	La macchina che legge: cos'è un LLM	23
2.1	Non è un database, non è Google, non è un robot	24
2.1.1	Non è un database	24
2.1.2	Non è Google	25
2.1.3	Non è un robot	26
2.1.4	Allora cos'è?	27
2.2	Come ha imparato: il training in parole semplici	29
2.2.1	Il compito apparentemente semplice che cambia tutto	29
2.2.2	Perché prevedere la parola successiva insegna tutto il resto	30
2.2.3	L'analogia giusta: non il pappagallo, il navigatore	30
2.2.4	Dal modello base all'assistente: il secondo strato di addestramento	31
2.2.5	La scala che cambia tutto	32
2.3	Token e contesto: come il modello «vede» il testo	32
2.3.1	Il token: l'unità minima di lettura	33
2.3.2	Il costo dell'italiano	33
2.3.3	La context window: il campo visivo del modello	34
2.3.4	Il problema «lost in the middle»: l'attenzione non è uniforme	34
2.3.5	La memoria che non c'è: stateless per design	35
2.3.6	La context window come risorsa da gestire	36
2.4	Il problema delle allucinazioni: perché inventa le risposte	36
2.4.1	Cosa sono esattamente le allucinazioni	36
2.4.2	Perché succede: la radice del problema	37
2.4.3	Le allucinazioni non sono tutte uguali	37
2.4.4	Il paradosso della sicurezza: perché il modello non sa di non sapere	38
2.4.5	Quando le allucinazioni sono un problema serio	39
2.4.6	Quando le allucinazioni sono gestibili	39
2.4.7	Le soluzioni architetturali: come ridurre le allucinazioni nei sistemi aziendali	40
2.4.8	Il quadro che conta davvero	40
2.5	I modelli che esistono oggi: orientarsi nel mercato	41
2.5.1	Come è strutturato il mercato: tre livelli da capire	41
2.5.2	I modelli proprietari: i tre grandi	42
2.5.3	I modelli open source: la seconda economia dell'AI	43
2.5.4	Le dimensioni di scelta che contano davvero	43
2.5.5	Una nota sul fine-tuning: quando ha senso e quando no	44
2.5.6	Come muoversi in un mercato che cambia ogni sei mesi	45
2.6	Primo contatto: cosa succede davvero quando chiami un'API	45

3	Parlare alla macchina: l'arte del prompt	47
3.1	Istruzioni per un collaboratore molto letterale	48
3.1.1	La metafora giusta: il collaboratore letterale	48
3.1.2	Il costo dell'ambiguità	49
3.1.3	Il problema dei task aperti	49
3.1.4	La qualità del prompt determina la qualità dell'output	50
3.1.5	Il contesto professionale implicito	50
3.1.6	Ma non sto già facendo il lavoro io?	51
3.1.7	Il prompting come competenza collettiva	51
3.2	Anatomia di un prompt efficace	52
3.2.1	Il ruolo — chi deve essere il modello	52
3.2.2	Il contesto — cosa sa già	53
3.2.3	Il task — cosa deve fare, esattamente	54
3.2.4	Il formato — come deve presentare la risposta	54
3.2.5	I vincoli — cosa non deve fare	55
3.2.6	I cinque componenti insieme: l'esempio prima e dopo	55
3.2.7	Sviluppare l'abitudine: il checklist mentale	57
3.3	Il system prompt: la voce aziendale sempre attiva	58
3.3.1	Cos'è il system prompt: la struttura tecnica spiegata senza tecnicismi	58
3.3.2	La differenza che fa: da modello generico ad assistente aziendale	59
3.3.3	Anatomia di un system prompt aziendale	59
3.3.4	Il system prompt come strumento di compliance e governance	61
3.3.5	System prompt e personalizzazione per contesto	62
3.3.6	Scrivere un system prompt: le regole pratiche	63
3.3.7	Un esempio completo: dalla prima bozza alla versione affinata	63
3.3.8	Il system prompt come decisione aziendale, non tecnica	64
3.4	Tecniche avanzate senza scrivere codice	65
3.4.1	Few-shot prompting: insegnare con gli esempi	65
3.4.2	Chain-of-thought: chiedere al modello di pensare ad alta voce	67
3.4.3	Prompt chaining: spezzare il grande nel piccolo	69
3.4.4	Le tre tecniche insieme: quando combinarle	72
3.4.5	Una nota finale: queste tecniche non richiedono tecnica	73
3.5	Gli errori più comuni: perché il modello non fa quello che vuoi	73
3.5.1	Errore 1 — Le istruzioni ambigue	74
3.5.2	Errore 2 — Il contesto insufficiente o assente	75
3.5.3	Errore 3 — Il task troppo ampio	76
3.5.4	Errore 4 — Il formato non dichiarato	77

3.5.5	Errore 5 — L'assenza di vincoli negativi	77
3.5.6	Errore 6 — Il prompt «one shot» su task iterativi	78
3.5.7	Errore 7 — Chiedere opinioni senza dare criteri	79
3.5.8	Errore 8 — Fidarsi dell'output senza verificarlo	80
3.5.9	Una mappa degli errori: dal sintomo alla causa	81
3.6	Prompt engineering vs. programmazione: cosa può fare un non-tecnico	82
3.6.1	Il confine reale tra prompting e programmazione	82
3.6.2	Chi ottiene i risultati migliori: il profilo che sorprende	83
3.6.3	Cosa si impara e in quanto tempo	83
3.6.4	Lo snippet: lo stesso task, tre prompt diversi	84
3.6.5	La mappa della differenza tra le tre versioni	87
3.6.6	Come sviluppare questa competenza nella pratica	88
3.6.7	Il punto di arrivo: da strumento a competenza organizzativa	88
4	Insegnare al modello: quando e perché specializzarlo	91
4.1	La domanda che tutti fanno	93
4.1.1	Perché la domanda è legittima — e da dove viene	93
4.1.2	Le tre preoccupazioni reali dentro la domanda	93
4.1.3	L'aspettativa che crea più problemi	94
4.1.4	Perché onorare la domanda — e risponderle seriamente	94
4.2	Cosa significa davvero fare fine-tuning	95
4.2.1	Il punto di partenza: cosa sa già il modello base	95
4.2.2	Il meccanismo: cosa succede durante il fine-tuning	96
4.2.3	Non esiste «un» fine-tuning: le varianti principali	96
4.2.4	Cosa entra: il dataset di training	97
4.2.5	Cosa esce: un modello diverso, non un modello aggiornato	98
4.2.6	Il processo completo: dalla decisione al deploy	99
4.2.7	Tre cose che il fine-tuning non fa	99
4.3	Quando il fine-tuning è la scelta giusta	101
4.3.1	Scenario 1 — Tono e stile aziendale come requisito non negoziabile	102
4.3.2	Scenario 2 — Task strutturati con output prevedibile e formato fisso	104
4.3.3	Scenario 3 — Ambienti chiusi senza accesso a internet o a API esterne	106
4.3.4	Scenario 4 — Latenza critica: quando ogni millisecondo conta	107
4.3.5	Il filo comune tra i quattro scenari	109
4.4	Quando il fine-tuning è un vicolo cieco	110
4.4.1	Difetto strutturale 1 — La conoscenza si congela al momento del training	110
4.4.2	Difetto strutturale 2 — Il paradosso della specificità: più specializzato, meno flessibile	112

4.4.3	Difetto strutturale 3 — Il costo nascosto: dati, annotazione, manutenzione continua	113
4.4.4	Il caso reale: quando tutti e tre i difetti colpiscono insieme	114
4.4.5	Il filo comune: quando il vicolo cieco è prevedibile	116
4.5	Il costo reale: quello che nessuno calcola prima di iniziare	117
4.5.1	Fase 1 — Definizione del task e dello scope	118
4.5.2	Fase 2 — Raccolta e cura dei dati	118
4.5.3	Fase 3 — Annotazione: la voce di costo più alta e più invisibile	120
4.5.4	Fase 4 — Training: la parte visibile, non la più costosa	121
4.5.5	Fase 5 — Valutazione: misurare quello che conta davvero	122
4.5.6	Fase 6 — Deploy e infrastruttura di serving	123
4.5.7	Fase 7 — Manutenzione nel tempo: il costo che nessuno pianifica	124
4.5.8	Il Total Cost of Ownership: come si accumula nel tempo	125
4.6	L'albero decisionale: fine-tuning sì o no?	126
4.6.1	L'albero decisionale	126
4.6.2	Il kit del manager: le domande da fare al team tecnico	128
4.6.3	Anatomia di un dataset di fine-tuning: cosa si sta commissionando	130
4.6.4	Il posizionamento finale: fine-tuning, RAG e prompting a confronto	131
5	La biblioteca esterna: il RAG	133
5.1	L'intuizione fondamentale: cerca prima, rispondi dopo	133
5.1.1	L'analogia del consulente e della biblioteca	134
5.1.2	Perché questa differenza cambia tutto	134
5.1.3	Un'obiezione legittima: il modello non capisce tutto	135
5.1.4	Il confine tra memoria e ricerca: un caso limite utile	136
5.1.5	Cosa il RAG non risolve	136
5.2	Come i documenti diventano materiale consultabile	137
5.2.1	Tappa 1 — Il chunking: tagliare senza spezzare il senso	137
5.2.2	Tappa 2 — Gli embedding: trasformare il significato in numeri	139
5.2.3	Tappa 3 — Il vector database: archiviare e cercare per vicinanza	141
5.2.4	Il metadato: il dettaglio che moltiplica l'utilità	141
5.2.5	Il ciclo completo di indicizzazione: da documento a chunk vettoriale	143
5.3	La ricerca semantica: trovare il senso, non la parola	144
5.3.1	Come funziona la ricerca tradizionale — e dove si rompe	144
5.3.2	La ricerca semantica: come funziona davvero	145
5.3.3	L'esempio completo: dal testo alla distanza semantica	146
5.3.4	Ricerca keyword vs. ricerca semantica: il confronto sistematico	146
5.3.5	La ricerca ibrida: quando si combinano i due approcci	146

5.3.6	Dove la ricerca semantica fatica: i limiti che contano	148
5.3.7	Il re-ranking: affinare i risultati dopo il recupero	148
5.3.8	Cosa succede alla query prima della ricerca: la query expansion	148
5.4	Dalla ricerca alla risposta: il ciclo completo	149
5.4.1	Il ciclo in astratto: sette passi	149
5.4.2	Il ciclo percorso in concreto: il caso Meridiana Assicurazioni	152
5.4.3	Il ciclo quando qualcosa va storto: un secondo scenario	154
5.4.4	Cosa i due scenari insegnano sulla governance del sistema	155
5.5	RAG vs. fine-tuning: la mappa decisionale	156
5.5.1	Otto dimensioni di confronto	156
5.5.2	Approfondimento dimensione per dimensione	158
5.5.3	La mappa degli scenari: quando usare cosa	160
5.5.4	L'architettura ibrida: non un compromesso, una scelta progettuale	161
5.5.5	Le domande da fare a un fornitore — o a un team interno	162
5.5.6	Una guida visiva alla decisione	163
6	La risposta che si fida dei dati: qualità e affidabilità	165
6.1	Il problema della demo perfetta	165
6.1.1	La demo è un ambiente progettato, non un ambiente reale	165
6.1.2	Poi arriva la produzione	166
6.1.3	Il problema specifico dell'AI: i fallimenti silenziosi	166
6.1.4	La trappola del successo iniziale	167
6.1.5	Cosa significa, in pratica, non fidarsi della demo	167
6.2	Cosa può andare storto in un sistema RAG	167
6.2.1	Fallimento 1 — Il sistema recupera i documenti sbagliati	168
6.2.2	Fallimento 2 — Il sistema recupera i documenti giusti, ma il modello li ignora .	169
6.2.3	Fallimento 3 — Il sistema risponde correttamente, ma in modo non verificabile	170
6.2.4	Fallimento 4 — Il sistema funziona bene nel linguaggio generale e male nei contesti tecnici specifici	171
6.2.5	Il quadro complessivo: perché questi quattro fallimenti si intrecciano	172
6.3	Le metriche tecniche che devi conoscere (anche se non le calcoli tu)	173
6.3.1	Faithfulness — La risposta è fedele ai documenti?	173
6.3.2	Answer Relevance — La risposta risponde davvero alla domanda?	174
6.3.3	Context Precision — I documenti recuperati sono davvero utili?	175
6.3.4	Il quadro integrato: come le tre metriche si parlano	176
6.3.5	Una parola sulla misurazione automatica versus quella umana	176
6.4	Le metriche di business che contano davvero	177

6.4.1	Deflection rate — Quante richieste gestisce l'AI senza intervento umano? . . .	178
6.4.2	Tempo medio di risoluzione — L'AI velocizza davvero?	179
6.4.3	Tasso di escalation — Quante volte l'AI chiede aiuto?	179
6.4.4	Soddisfazione utente — Le persone si fidano del sistema?	180
6.4.5	Come costruire un caso di ROI onesto	181
6.5	Il gap tra la demo e la produzione: cosa aspettarsi	182
6.5.1	Fase 1 — Il PoC: la promessa controllata	182
6.5.2	Fase 2 — Il pilot: dove la realtà inizia a collaborare	183
6.5.3	Fase 3 — La produzione: dove i problemi veri si rendono visibili	184
6.5.4	La regola dei tre budget — e perché nessuno la rispetta	185
6.5.5	Quando fermarsi, quando accelerare	185
6.6	Le domande da fare al tuo team tecnico	186
6.6.1	Prima di approvare il progetto	186
6.6.2	Durante lo sviluppo	187
6.6.3	Prima del go-live	188
6.6.4	Una nota finale su come usare queste domande	189
7	Oltre la risposta: il sistema che agisce	191
7.1	Il limite del sistema che risponde	191
7.1.1	Il paradosso dell'intelligenza disarmata	192
7.1.2	Dove il limite era accettabile — e dove ha smesso di esserlo	192
7.1.3	Il costo nascosto che nessuno contabilizza	193
7.1.4	Quando il sistema che risponde è la scelta giusta	194
7.2	Cos'è un agente: la definizione che conta	194
7.2.1	La definizione operativa	194
7.2.2	Il ciclo percezione–ragionamento–azione: un caso aziendale completo	195
7.2.3	La differenza con i sistemi precedenti: un confronto preciso	197
7.2.4	Il confine dell'autonomia: il requisito più importante di tutti	197
7.3	Gli strumenti dell'agente: come tocca il mondo reale	198
7.3.1	L'analogia del nuovo dipendente	198
7.3.2	Cos'è concretamente uno strumento	199
7.3.3	Il catalogo degli strumenti: un dizionario di azioni possibili	200
7.3.4	Strumenti per categoria: cosa esiste negli ecosistemi aziendali	200
7.3.5	Perché la qualità degli strumenti è critica quanto la qualità del modello	201
7.3.6	Gli strumenti che non dovrete mai dimenticare di includere	202
7.4	Da chatbot ad agente: cosa cambia in pratica	203
7.4.1	Il problema: una richiesta di reso complicata	203

7.4.2	Il chatbot classico: la risposta elegante che non risolve niente	203
7.4.3	L'agente: stessa situazione, percorso radicalmente diverso	204
7.4.4	La differenza non è nella risposta: è nell'architettura del valore	206
7.4.5	Il ruolo che rimane — e quello che dovrebbe rimanere — all'umano	206
7.4.6	I segnali che indicano che siete pronti per il salto	207
7.5	I casi d'uso che funzionano oggi	207
7.5.1	Scenario 1 — Customer service: gestione autonoma dei reclami standard . . .	207
7.5.2	Scenario 2 — Sales enablement: l'assistente commerciale che qualifica e prepara	209
7.5.3	Scenario 3 — IT helpdesk: risoluzione autonoma dei ticket di primo livello . .	210
7.5.4	Il filo rosso tra i tre scenari	211
7.6	Primo agente: il codice che apre una porta	211
7.6.1	Il caso: un agente che gestisce i ticket di supporto IT	211
7.6.2	Cosa sta succedendo, riga per riga	219
7.6.3	Le domande da fare al vostro team dopo aver letto questo codice	220
8	Dare potere senza perdere controllo: guardrail e supervisione	221
8.1	Il problema dell'autonomia: cosa può andare storto	221
8.1.1	L'agente che interpreta troppo liberamente	222
8.1.2	L'agente che esegue un'azione irreversibile senza conferma	222
8.1.3	L'agente che ottimizza la metrica sbagliata	223
8.1.4	Il pattern comune: il sistema ha fatto quello che gli era stato chiesto	224
8.2	Il principio del minimo privilegio: dare solo il potere necessario	225
8.2.1	Cosa significa «minimo privilegio» per un agente AI	225
8.2.2	Progettare il perimetro: dall'obiettivo agli strumenti	225
8.2.3	Il paradosso della facilità di sviluppo	226
8.2.4	Il minimo privilegio come conversazione organizzativa	227
8.3	I tre livelli di controllo: prima, durante, dopo	227
8.3.1	Livello 1 — Controlli sull'input: cosa entra nel sistema	228
8.3.2	Livello 2 — Controlli durante l'esecuzione: cosa sta facendo il sistema	229
8.3.3	Livello 3 — Controlli sull'output: cosa ha prodotto il sistema	230
8.3.4	Il framework integrato: come i tre livelli si parlano	231
8.4	Human-in-the-loop: quando l'uomo deve restare nel circuito	231
8.4.1	Il continuum dell'autonomia	231
8.4.2	Quando il human-in-the-loop è necessario	232
8.4.3	Progettare la supervisione in modo che funzioni davvero	233
8.5	Audit trail e trasparenza: sapere cosa ha fatto il sistema	233
8.5.1	Perché l'audit trail è diverso dal log tecnico	234

8.5.2	Cosa deve contenere un audit trail per un sistema agente	234
8.5.3	La sfida della spiegabilità	235
8.5.4	Audit trail come strumento manageriale, non solo legale	236
8.6	Un framework decisionale: quanto autonomia dare?	236
8.6.1	Le tre variabili	236
8.6.2	Lo schema decisionale	237
8.6.3	Il frammento tecnico: come si definisce un tool con validazione e limiti	238
8.6.4	Il framework come documento, non come intuizione	241
8.6.5	La governance come competitività	241
9	Più agenti insieme: quando un solo sistema non basta	243
9.1	Il limite del solista: perché un agente non basta	243
9.1.1	La trappola dell'agente generalista	243
9.1.2	Il parallelo organizzativo: perché esistono i dipartimenti	244
9.1.3	Quando il task attraversa i confini	244
9.1.4	Il costo della complessità prematura	245
9.2	L'orchestra: come gli agenti collaborano	245
9.2.1	L'architettura dell'orchestrazione	245
9.2.2	Come fluisce l'informazione tra agenti	246
9.2.3	L'esecuzione in parallelo: quando la velocità moltiplica il valore	247
9.2.4	La questione della fiducia tra agenti	248
9.3	Cosa cambia quando gli agenti sono molti	248
9.3.1	Il problema del coordinamento: chi decide e come	248
9.3.2	Il problema del fallimento parziale: cosa succede quando un agente cade	249
9.3.3	Il problema della qualità aggregata: gli errori si moltiplicano o si compensano?	249
9.3.4	Il problema della governance distribuita: chi è responsabile di cosa?	250
9.3.5	Il problema dell'emergenza: comportamenti che nessuno ha progettato	251
9.3.6	Il problema della spiegabilità amplificata	251
9.4	Tre casi reali: il multi-agente che crea valore oggi	251
9.4.1	Caso 1 — Customer journey nell'e-commerce: da visitatore a cliente risolto	252
9.4.2	Caso 2 — Due diligence legale automatizzata: il lavoro di settimane in ore	253
9.4.3	Caso 3 — Procurement e supply chain: l'intelligenza nella catena di fornitura	254
9.5	Il viaggio è finito. O forse è appena cominciato	256
9.5.1	Quello che sapete ora — e perché conta	256
9.5.2	Cosa non sapete ancora — e dove si trova	257
9.5.3	Il frammento finale: due agenti che si passano un task	257
9.5.4	Una nota finale sul prossimo viaggio	259

