TECHNOLOGY & INNOVATION

# L'Intelligenza Artificiale nel ciclo di sviluppo del Software

ASSIST DIGITAL

# Executive Summary

Le aziende di IT services operano sotto una pressione crescente: innovare più velocemente, garantire qualità e sicurezza del software su larga scala e gestire margini sempre più ridotti. Per rispondere a queste sfide, l'adozione dell'Intelligenza Artificiale nel Software Development Lifecycle (SDLC) non è più un'opzione esplorativa ma una necessità strategica. Nel 2026, circa l'80%[1] degli sviluppatori utilizza strumenti AI in almeno una fase del ciclo di vita del software, e il 40%[2] del codice globale è già generato o co-generato dall'AI. Tuttavia, la maggior parte delle organizzazioni rimane in una fase di adozione parziale, concentrata su use case isolati piuttosto che su una trasformazione sistemica.

Il presente documento nasce dalla sinergia tra l'esperienza diretta maturata nel progetto AI Adoption della unit CX Technology di Assist Digital e un'approfondita analisi di report e studi di mercato di terze parti. Il documento analizza l'impatto dell'Intelligenza Artificiale su processi, tecnologie, benefici e rischi, offrendo una visione d'insieme sull'evoluzione organizzativa necessaria per governare il cambiamento.

## Evidenze chiave

**L'AI ridisegna l'intero SDLC, non solo il coding.** L'impatto non è limitato alla code generation: dall'analisi semantica dei requisiti al design architetturale, al testing, ogni fase subisce una trasformazione qualitativa. La distinzione tra fasi sequenziali sfuma in favore di un flusso agentico fluido, con tempi di completamento che passano da giorni a ore.

**Il guadagno di produttività è reale ma richiede un modello di indirizzo e controllo efficace.** I dati aggregati indicano un risparmio tra il 30% e il 60%[8] su coding, testing e documentazione. Tuttavia, il 62%[9] degli sviluppatori segnala rischi di debito tecnico. Assist Digital ha mitigato questo "Paradosso GenAI" integrando l'AI direttamente nelle pipeline CI/CD con quality gate automatizzati e una cultura "human-owned" in cui la responsabilità resta umana.

**L'agentic coding è il vero punto di svolta.** Il passaggio da semplici assistenti di autocompletamento a sistemi agentici capaci di gestire task multi-step in modo semi-autonomo/autonomo è la novità più dirompente del 2025-2026. Gli sviluppatori utilizzano strumenti AI in circa il 60%[5] delle attività, anche se la delega completa resta ad un numero limitato di task.

**L'impatto organizzativo è profondo e non rinviabile.** L'AI non elimina i ruoli tecnici ma li trasforma: i developer evolvono verso orchestrazione e validazione, i senior developer verso architettura e governance AI. Il modello di staffing dinamico per pools tecnologici adottato da Assist Digital ottimizza l'utilization gestendo più servizi con lo stesso team potenziato dall'AI.

**Non il tool, ma il processo di adozione determina il risultato.** L'AI agisce come "mirror and multiplier": amplifica l'efficienza dove esistono pratiche solide, amplifica le debolezze dove manca metodo. Il progetto Assist Digital ha strutturato l'adozione con baseline KPI pre-AI, scaling graduale, evangelist di area e dashboard ROI continuativa, dimostrando che il vantaggio competitivo nasce dalla disciplina del processo, non dalla potenza del singolo strumento.

# index


01 Come l’AI sta cambiando i processi di sviluppo software

02 Le fasi del ciclo di sviluppo: verso l’ecosistema multi-agente

03 Le tecnologie AI più utilizzate e le tendenze

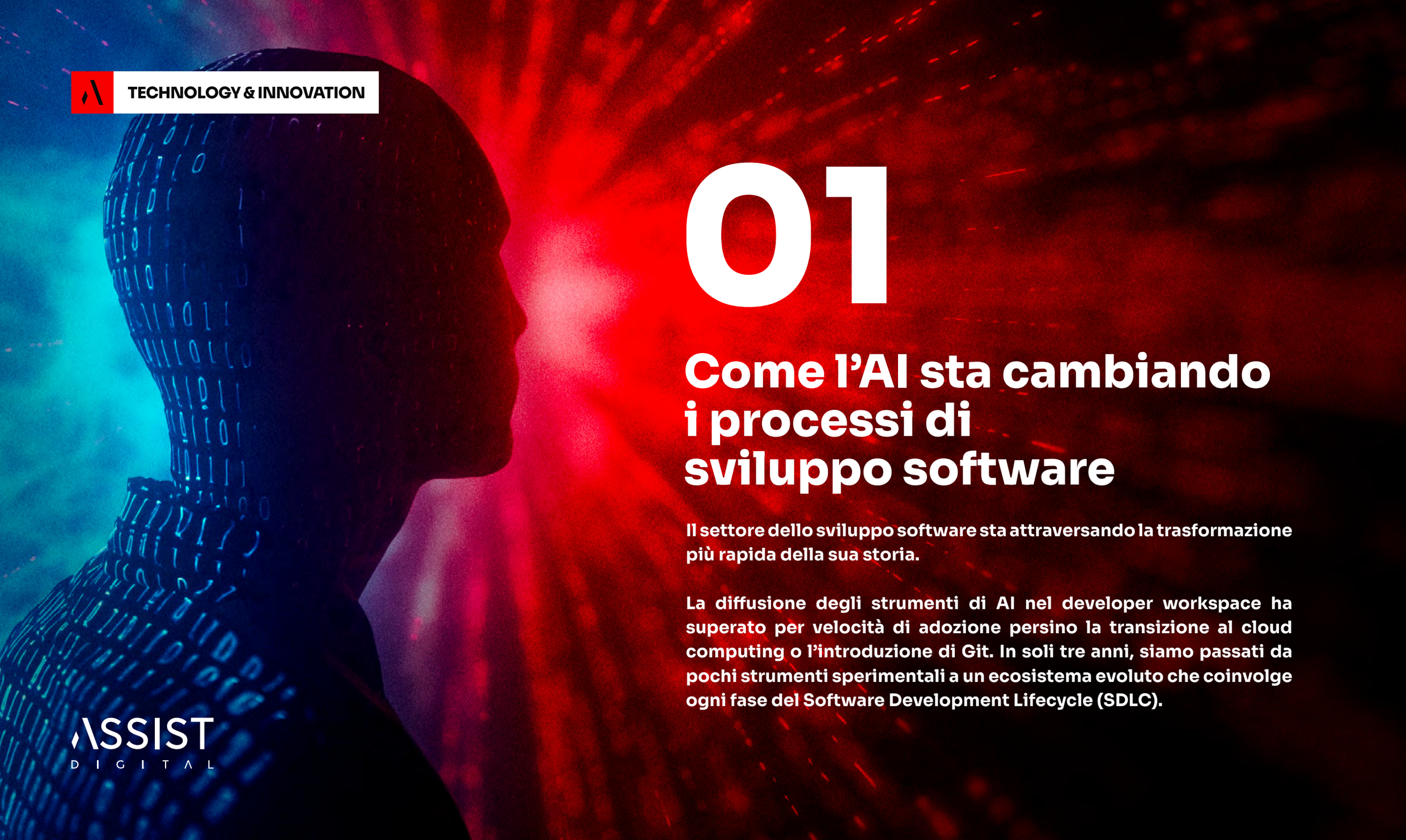
04 Benefici strategici e mitigazione del “Paradosso GenAI”

05 Impatto organizzativo e staffing dinamico

**L’adozione dell’Intelligenza Artificiale nello sviluppo software non è più una tendenza futura: è una trasformazione in atto.**

Nel 2026, circa l’80%[1] degli sviluppatori utilizza strumenti AI in almeno una fase del ciclo di vita del software e circa il 40%[2] del codice globale è già generato o co-generato dall’AI.

Questo documento analizza l’impatto di questa rivoluzione su processi, tecnologie, benefici, rischi e organizzazione, con una prospettiva maturata dall’esperienza diretta di Assist Digital nel progetto interno di AI Adoption per il team CX Technology.



TECHNOLOGY & INNOVATION

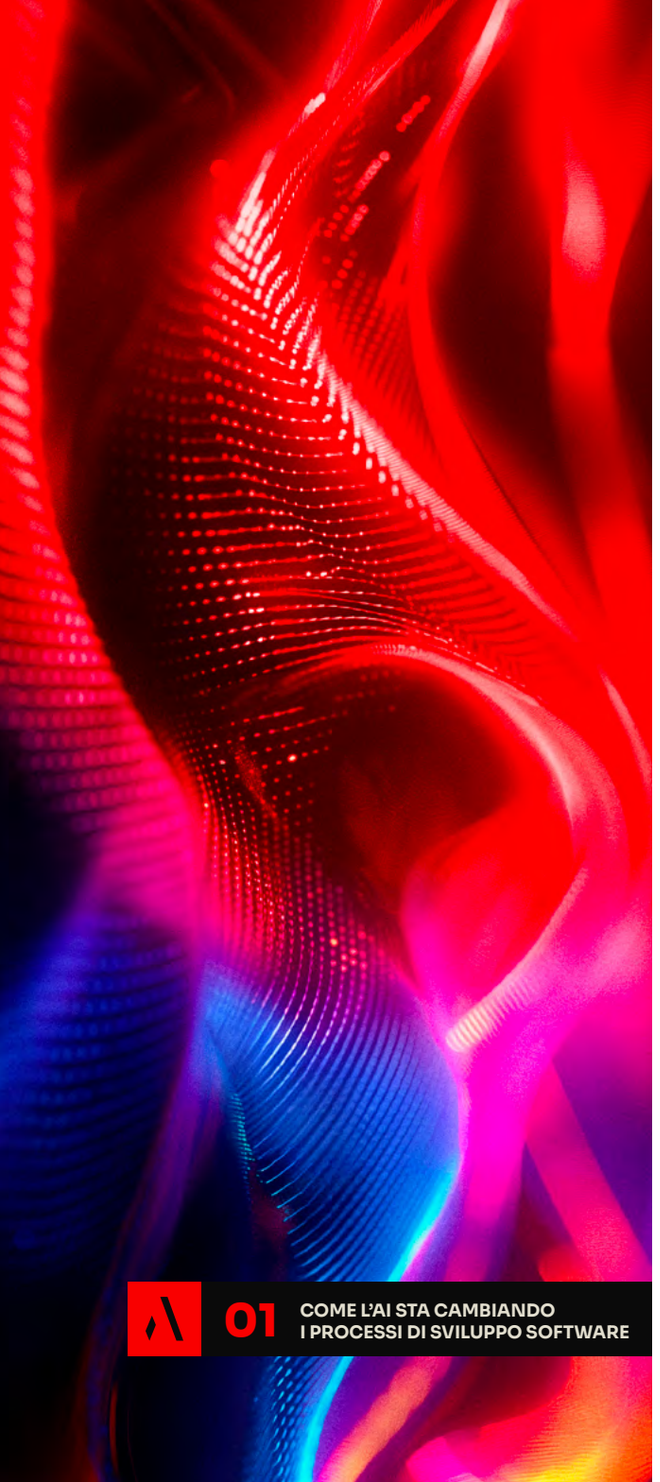
# 01

# Come l'AI sta cambiando i processi di sviluppo software

**Il settore dello sviluppo software sta attraversando la trasformazione più rapida della sua storia.**

**La diffusione degli strumenti di AI nel developer workspace ha superato per velocità di adozione persino la transizione al cloud computing o l'introduzione di Git. In soli tre anni, siamo passati da pochi strumenti sperimentali a un ecosistema evoluto che coinvolge ogni fase del Software Development Lifecycle (SDLC).**

ASSIST DIGITAL




# Il punto di svolta: dall'assistenza all'autonomia

Fino al 2023, l'utilizzo dell'AI nel ciclo di sviluppo del software era essenzialmente uno strumento di autocompletamento avanzato. Oggi siamo di fronte a una trasformazione qualitativa: i Large Language Models (LLM) si sono evoluti in sistemi agentici capaci di comprendere requisiti in linguaggio naturale, generare architetture, scrivere test, effettuare code review e gestire incidenti in produzione con supervisione umana ridotta.

Il concetto di "agentic coding", ovvero la capacità degli strumenti AI di eseguire task complessi in modo semi-autonomo/autonomo, iterando su feedback e collegando più step in sequenza, è la novità più dirompente del 2025-2026. Secondo il report di Anthropic "2026 Agentic Coding Trends", stiamo assistendo a un ridisegno sistemico dell'intero SDLC, con attività che richiedevano settimane di coordinamento tra team che diventano sessioni di lavoro concentrate.

# Le fasi del SDLC: prima e dopo l'AI

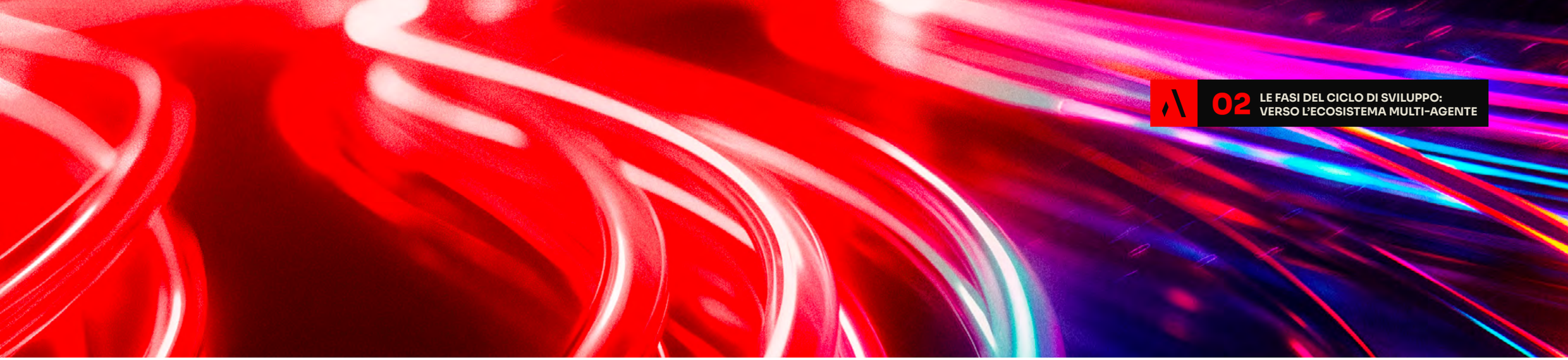
L'impatto dell'AI non è uniforme lungo tutto il ciclo di vita del software. Alcune fasi vengono profondamente trasformate, altre più gradualmente. In termini generali, le attività più codificate, ripetitive e basate su pattern sono quelle in cui l'AI esprime il massimo potenziale; le attività che richiedono giudizio contestuale, comprensione del business e creatività restano a forte presidio umano, almeno nel breve termine.

La narrativa della "sostituzione" del developer ha lasciato spazio a una visione più sfumata e realistica: l'AI come moltiplicatore di capacità, non come sostituto. Il CEO di Google, Sundar Pichai, ha sintetizzato questa visione: il 25% del codice di Google è già AI-assistito, ma la metrica che conta è l'aumento della velocità ingegneristica complessiva, stimata intorno al 10%, non la riduzione di sviluppatori.

# 02

# Le fasi del ciclo di sviluppo: verso l'ecosistema multi-agente

**L'impatto dell'AI sul ciclo di sviluppo non si esaurisce in un guadagno di velocità isolato: ridisegna il modello di delivery. Le fasi del SDLC, storicamente sequenziali, convergono in un flusso agentico più fluido, in cui task che richiedevano giorni o settimane possono concludersi nell'ordine di ore.**





# Analisi e Design: dalla raccolta alla generazione

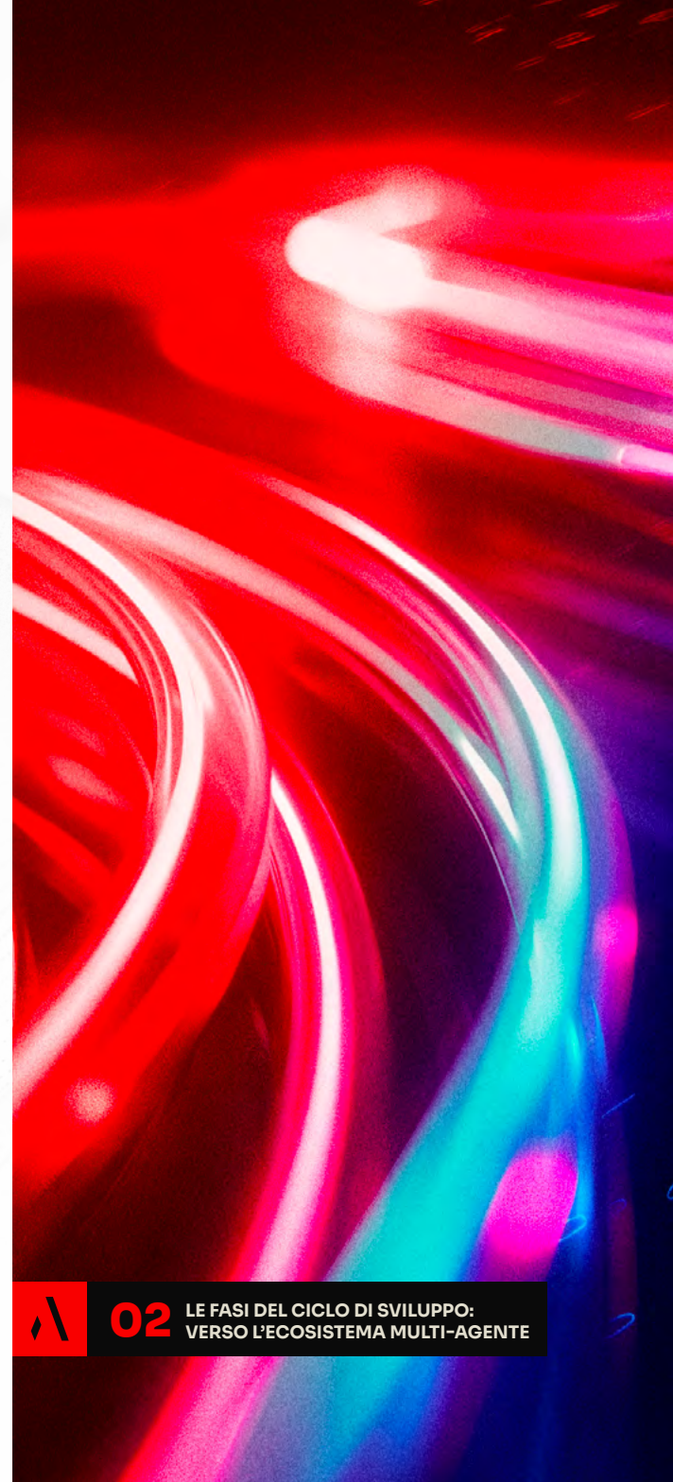
L'adozione di strumenti di agentic coding ha permesso di trasformare l'analisi funzionale da attività labour-intensive a un processo di sintesi intelligente:

- Analisi semantica: estrazione automatica di requisiti strutturati da transcript e documenti informali, riducendo le ambiguità prima dello sviluppo.
- Aumento di produttività: incremento percepito tra il 20% e il 30%[3] nella produzione di user stories ed epiche.
- Design architetturale: l'intelligenza artificiale agisce come un moltiplicatore di efficacia per gli architetti[4], supportando la definizione di stack tecnologici e la simulazione della scalabilità attraverso pratiche di Platform Engineering mature. Questo consente ai profili senior di evolvere verso ruoli di supervisione strategica e governance architetturale.

# Sviluppo Software (Coding)

La fase di coding è la più impattata, con il 40%[2] del codice globale co-generato dall'AI. È la fase più impattata in termini assoluti e quella che ha ricevuto maggiore attenzione mediatica. L'AI nel coding si manifesta su più livelli:

- Autocompletamento avanzato e code generation: da semplici snippet a intere funzioni o moduli.
- Debugging intelligente: identificazione di bug in fase di scrittura con suggerimento di correzioni.
- Code review automatizzata: revisioni di pull request per stile, sicurezza e conformità agli standard.
- Refactoring e ottimizzazione: suggerimento di riscritture per migliorare performance e manutenibilità.
- Agentic Coding: esecuzione autonoma di task multi-step, dall'analisi del ticket alla scrittura, test e documentazione del codice.



# Sviluppo Software (Coding)

Il dato più emblematico: secondo il report di Anthropic “2026 Agentic Coding Trends”[5], gli sviluppatori utilizzano strumenti di AI in circa il 60% delle loro attività, ma solo una quota limitata dei task, sotto il 20% circa, viene attualmente delegata completamente agli agenti.

**GLI SVILUPPATORI UTILIZZANO STRUMENTI DI AI IN CIRCA IL 60% DELLE LORO ATTIVITÀ.**

**60%**

# Testing e Quality Assurance

Il testing è tradizionalmente una delle fasi più cruciali del ciclo di vita dello sviluppo del software. L'AI introduce una trasformazione profonda con:

- Generazione automatica di test case, inclusi scenari difficili da immaginare manualmente.
- Test predittivi: identificazione delle aree di codice a maggiore rischio di regressione su cui concentrare il testing.
- Creazione e gestione di suite di test di regressione e stress test.
- Gestione intelligente del rilascio: scelta ottimale delle finestre temporali per il deploy.

Nel progetto Assist Digital, il team di Quality Assurance utilizza sistemi di agentic coding per la scrittura ed esecuzione di test, con un impatto sulla copertura del codice e una riduzione stimata del 10-15%[6] del tempo di testing.

# Operations, monitoring e manutenzione

L'AI applicata alle attività di IT Operations, spesso indicata con il termine AIOps, è la frontiera più matura dopo il coding. I sistemi di AI applicati alle IT Operations permettono:

- Monitoraggio delle performance in tempo reale con rilevamento anomalie e alert predittivi.
- Correlazione automatica di eventi IT per identificare root cause di incidenti.
- Analisi automatica dei ticket di manutenzione: classificazione, prioritizzazione e suggerimento di soluzioni.
- Supporto alla manutenzione correttiva: analisi dei log, correlazione con il codice sorgente, suggerimenti per il bug fixing.

Quest'ultima capacità è particolarmente strategica per le aziende di IT services come Assist Digital: l'AI può analizzare la segnalazione, correlare con la storia del codice e con i log applicativi, e proporre allo sviluppatore le possibili cause e le correzioni, riducendo sensibilmente il MTTR (Mean Time to Repair).

# 03

# Le tecnologie AI più utilizzate e le tendenze

ASSIST DIGITAL

# Il panorama tecnologico attuale

Il mercato degli strumenti AI per lo sviluppo software è in rapida crescita: da un valore stimato di 933 milioni di dollari nel 2025, si prevede raggiungerà i 15,7 miliardi entro il 2033, con un CAGR del 42,3%[7].

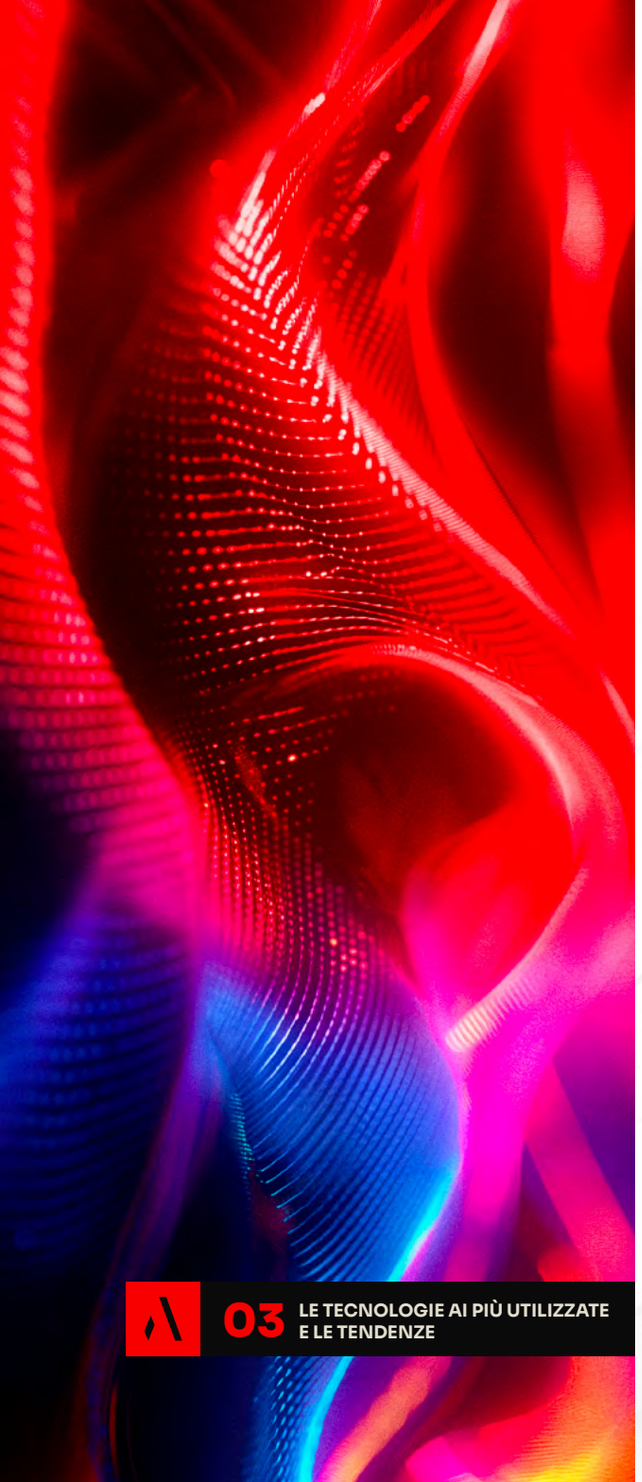
**15,7 mld**

Crescita del mercato AI per lo sviluppo software entro il 2023

## Large Language Models (LLM) di frontiera

Claude (Anthropic), GPT (OpenAI) e Gemini (Google) rappresentano la triade principale, con famiglie di modelli pensate per casi d'uso differenti - dal ragionamento multi-step ad alta complessità (es. Claude Opus, GPT-5.5) all'integrazione real-time negli ambienti di sviluppo. La scelta tra modelli dipende da specifiche caratteristiche: capacità di ragionamento multi-step, capacità agentiche, tool use, ampiezza della context window, velocità di risposta e costo per token.

Nel progetto Assist Digital, la scelta è ricaduta su un mix di agenti coding come strumento principale per attività cognitive complesse (coding, analisi, architettura, testing), per la sua eccellenza nel ragionamento multi-step, nelle capacità agentiche e nella qualità della generazione di codice, abbinato a strumenti conversazionali di GenAI, già disponibili nell'ecosistema IT aziendale, per attività general-purpose quotidiane.





## Coding assistants e IDE

GitHub Copilot e Claude Code sono i tool che stanno emergendo con il più alto tasso di adozione, ma il mercato si è frammentato con l'emergere di soluzioni specializzate. La tendenza dominante è l'integrazione diretta di questi strumenti negli IDE (Integrated Development Environments): gli strumenti non sono più applicazioni standalone ma co-pilot integrati nel flusso di lavoro, con accesso al contesto del codebase completo.

## Orchestrazione di workflow e automazione

La seconda frontiera è l'automazione dei processi attraverso orchestratori di workflow AI. Qui possiamo citare ad esempio n8n, una piattaforma di orchestrazione che permette di creare workflow trigger-based con utilizzo AI on-demand, integrata con i sistemi aziendali esistenti (i.e. CRM, Ticketing, Monitoring, CI/CD, ambienti Cloud).

## Trend emergenti: Agentic AI e Multi-Agent Systems

La tendenza più significativa per il 2026 è il passaggio da sistemi a singolo agente ad architetture multi-agente. In questi sistemi, agenti specializzati cooperano: uno analizza i requisiti, un secondo genera il codice, un terzo esegue i test, un quarto documenta, tutto in modo coordinato e semi-autonomo/autonomo.

Il risultato pratico è già visibile: Amazon ha stimato un risparmio di 4.500 developer-years e 260 milioni di dollari in un singolo progetto di migrazione utilizzando AI coding integrata nell'infrastruttura organizzativa, non come strumento individuale.

# Il framework tecnologico di Assist Digital

Il progetto AI Adoption di CX Technology ha organizzato gli strumenti AI su quattro livelli, che vanno dal più sofisticato al più immediato.
Ogni livello risponde a esigenze diverse, ma lavorano insieme.

Al primo livello ci sono gli strumenti di agentic coding: ambienti di sviluppo in cui agenti AI svolgono in autonomia interi compiti, dall'analisi di una richiesta fino alla scrittura, al test e alla documentazione del codice, sempre con una persona che supervisiona. Due elementi rendono questi strumenti davvero potenti. Il primo sono le Skills: istruzioni riutilizzabili che racchiudono le buone pratiche, le scelte tecniche e gli standard dei nostri progetti, così che l'agente sappia "come si lavora qui". Il secondo sono i connettori MCP (Model Context Protocol), che permettono all'agente di dialogare in modo ordinato con i nostri sistemi e con quelli esterni: repository di codice, ambienti cloud, sistemi di ticketing, monitoraggio, documentazione. Messi insieme, Skills condivise e connettori MCP trasformano l'agentic coding da strumento personale a risorsa condivisa di tutto il team.

Al secondo livello c'è un assistente conversazionale GenAI, usato ogni giorno da tutti per attività di uso comune: riassumere documenti, fare brainstorming, scrivere testi, analizzare informazioni al volo.

Al terzo livello c'è una piattaforma di automazione dei flussi di lavoro, che permette di creare processi automatici attivati da eventi, con l'AI che interviene quando serve, e collegata ai sistemi che già usiamo (Service Management, reporting, classificazione dei ticket).

Al quarto livello, infine, i modelli linguistici e sistemi agentici vengono integrati direttamente negli ambienti di sviluppo. In questo modo l'AI non è un'applicazione a parte, ma una componente naturale del lavoro quotidiano degli ingegneri.

Questa struttura a più livelli ci permette di bilanciare potenza, costo e controllo dei dati, senza legarci a un unico fornitore.


TECHNOLOGY & INNOVATION
04
Benefici strategici
e mitigazione del
“Paradosso GenAI”
ASSIST
DIGITAL

# Un business case orientato al valore

## Produttività e velocità di delivery

L’impatto sulla produttività è la variabile più citata e più dibattuta. I dati aggregati indicano un incremento tra il 30% e il 60%[8] nel tempo risparmiato su coding, testing e documentazione. Questa percezione trova riscontro anche nella nostra esperienza diretta: quando si confrontano i risultati con i feedback dei responsabili delle diverse aree di sviluppo, il tempo di realizzazione non è quasi mai il primo beneficio che citano. Quello che emerge con più forza è la capacità di prendere decisioni migliori, più informate, più rapide, più vicine ai dati, e il miglioramento della collaborazione tra team che prima lavoravano in silos. La produttività del singolo sviluppatore viene riconosciuta, ma quasi come conseguenza naturale di un modo diverso di lavorare, non come il punto di partenza. È un segnale importante: il valore dell’AI nello SDLC non si esaurisce nel guadagno di produttività individuale e le organizzazioni che lo capiscono prima sono quelle che ne traggono il beneficio più duraturo.

## Qualità del codice e riduzione dei difetti

Con l’AI che suggerisce correzioni in real-time durante la scrittura, i bug vengono intercettati prima di raggiungere il repository. L’analisi automatica delle pull request riduce il carico di lavoro necessario per la revisione manuale del codice, ampliando allo stesso tempo la copertura dei controlli. Il risultato atteso è una riduzione del numero di difetti in produzione e una migliore copertura dei test.

## Democratizzazione delle competenze

L’AI permette agli sviluppatori di operare oltre la propria area di specializzazione: un backend developer può affrontare task front-end con supporto AI, un analista funzionale può produrre documentazione tecnica di qualità, un operatore di service management può automatizzare flussi complessi senza competenze di programmazione. Questo effetto di “full-stack democratization” aumenta la flessibilità organizzativa.

**30%**

**AI**

**TEMPO RISPARMIATO SU CODING, GRAZIE ALL’UTILIZZO DI STRUMENTI AI[8]**

**60%**

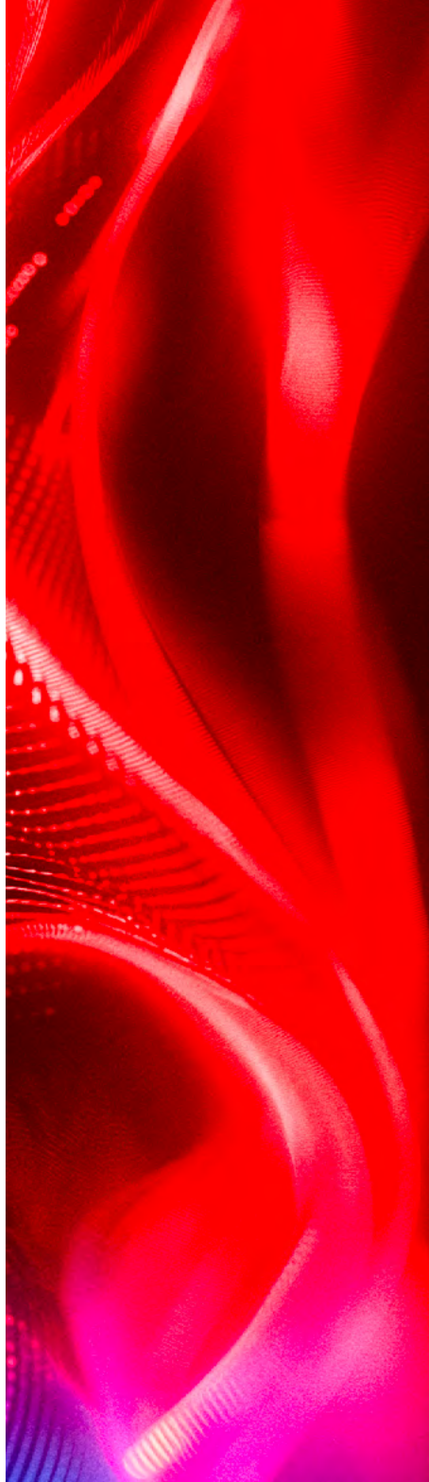
# Risolvere le criticità: l'approccio Assist Digital

L'adozione dell'AI nello sviluppo software porta con sé un paradosso concreto: si accelera la produzione di codice, ma senza i giusti presidi si rischia di accumulare debito tecnico alla stessa velocità. Più codice generato significa più superficie da manutenere, più potenziali vulnerabilità, più inconsistenze architetturali.  Per mitigare questo rischio, Assist Digital ha strutturato l'adozione su tre leve complementari.

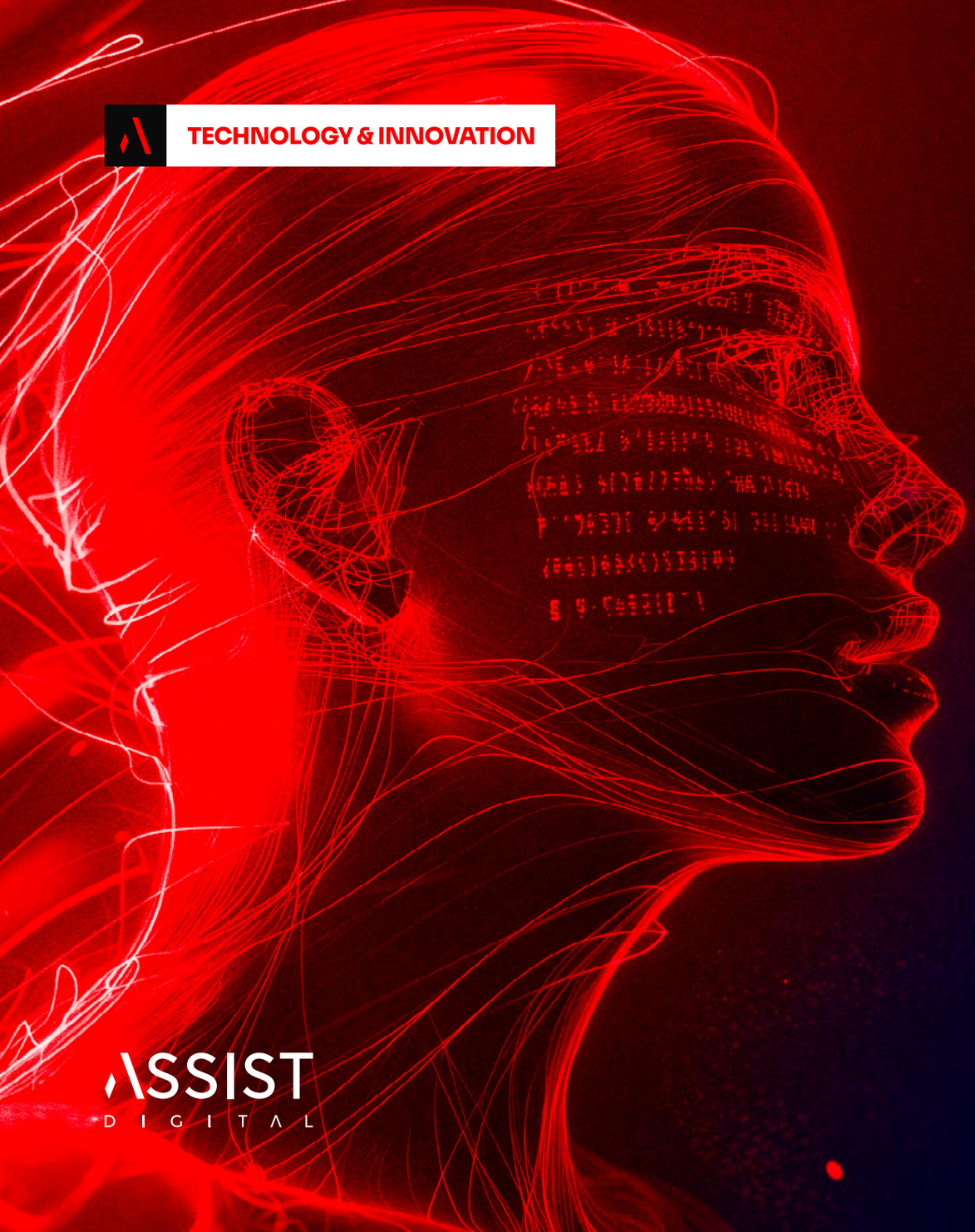
- **La prima è il rafforzamento sistematico dei processi di integrazione e rilascio continui.** Il codice generato dall'AI viene sottoposto agli stessi controlli, e in alcuni casi a controlli aggiuntivi, rispetto al codice scritto manualmente: controlli automatici di qualità, analisi statica del codice, verifiche di sicurezza e livelli di copertura dei test superiori rispetto alle soglie precedenti all'introduzione dell'AI. Il principio è semplice: se l'AI produce più codice, servono più controlli, non meno. Sebbene il 62%[9] degli sviluppatori segnali rischi di debito tecnico associati all'uso dell'AI, l'integrazione di tali presidi direttamente nel flusso di rilascio, anziché come attività successive, consente di individuare e correggere le criticità prima che il codice venga integrato nel repository.

- **La seconda leva è il ruolo dei profili senior.** Sviluppatori esperti e Architetti diventano i supervisori degli artefatti generati dall'AI: non scrivono meno codice perché l'AI lo fa al posto loro, ma investono il tempo liberato nella revisione critica, nell'aggiornamento costante delle best practice di utilizzo dell'AI e nel rafforzamento delle metodologie di progetto. È una transizione da "scrittore di codice" a "garante della coerenza architetturale e della qualità complessiva del sistema". Senza questo presidio, il rischio è che l'AI produca soluzioni localmente corrette ma globalmente incoerenti, poco sicure e tanto-meno manutenibili.

- **La terza è la cultura human-owned**: la responsabilità di ciò che va in produzione resta umana, sempre, e per questo lo deve restare anche la decisione. Una persona non può rispondere di una scelta che non ha realmente fatto. La diffidenza verso quanto produce l'AI non viene scoraggiata ma valorizzata come meccanismo di controllo naturale. L'AI è un assistente potente, non un decisore autonomo, e l'organizzazione è strutturata perché nessuno possa trattarla come tale.



TECHNOLOGY & INNOVATION



# 05

# Impatto organizzativo e staffing dinamico

ASSIST DIGITAL

# L’evoluzione dei ruoli

L’AI non elimina i ruoli tecnici ma li trasforma radicalmente, elevando ogni profilo verso attività di più alto valore.

- Evoluzione dei ruoli: i developer si focalizzano sulla validazione e sull’orchestrazione dei tool; i senior developer e gli architect diventano i garanti della coerenza architetturale e della security oversight.
- Pools tecnologici: ottimizzazione dell’utilization, perché lo stesso team diventa più versatile, capace di lavorare su stack tecnologici che prima avrebbero richiesto competenze separate.
- Innovazione continua: le risorse liberate dai task ripetitivi vengono reinvestite in attività di R&D per lo sviluppo di nuove offerte scalabili.

# Le competenze emergenti

Un numero sempre maggiore di aziende sta investendo in AI, ma lamenta una carenza di competenze che frena l'adozione. Report recenti sottolineano come, con l'AI che automatizza compiti ripetitivi, divengano prioritarie le skill umane come giudizio critico, comunicazione, leadership e comprensione del business.

AI literacy: comprensione del funzionamento dei modelli, dei loro limiti, delle tecniche di prompting efficace e delle best practice di supervisione degli output.

Agentic workflow design: capacità di progettare pipeline di automazione che combinano AI, sistemi aziendali e intervento umano nel punto giusto.

Quality & Security oversight: skill per valutare criticamente il codice AI-generato, identificare vulnerabilità di sicurezza e gestire il debito tecnico indotto dall'AI.

Sul mercato del lavoro, l'impatto è già visibile: PwC documenta che i professionisti con competenze AI avanzate guadagnano fino al 56%[10] in più rispetto ai colleghi dello stesso ruolo senza tali competenze.

# La piramide organizzativa

L'impatto sulla struttura organizzativa è uno degli aspetti più dibattuti e delicati. Due dinamiche contrastanti sono in atto simultaneamente:

## Compressione della base della piramide

Le attività tradizionalmente associate ai profili junior (scrittura di codice standardizzato, debugging semplice, testing manuale, documentazione) sono quelle più automatizzabili. I dati lo confermano: l'assunzione di profili entry-level in tecnologia è diminuita del 25% anno su anno nel 2024[11]. Uno studio di Stanford evidenzia un calo del 20%[12] nell'occupazione degli sviluppatori software tra 22 e 25 anni rispetto al picco del 2022. Più del 50%[13] dei developer senior prevede una riduzione dei task entry-level, con team più snelli e specializzati.

## Espansione del middle layer e della governance

AI contrario, crescono i ruoli di oversight, architettura e governance AI. Emergono figure come l'AI Engineer, AI Architect focalizzati su AI infrastructure e il Security AI Specialist. Gartner prevede che l'AI genererà intere nuove categorie di ruoli in software engineering e operations che non esistevano cinque anni fa.

### L'approccio di Assist Digital: staffing dinamico per pools tecnologici

La risposta di Assist Digital alle trasformazioni organizzative si articola su tre leve strategiche. **Prima**: il ridisegno dei ruoli, i developer evolvono da scrittori di codice a orchestratori di strumenti AI e validatori di output, mentre i senior assumono la responsabilità della coerenza architetturale, della security oversight e della supervisione dei sistemi multi-agente. **Seconda**: la formazione continuativa, in un contesto in cui strumenti, modelli e best practice dell'AI applicata allo SDLC si aggiornano a cicli di settimane, non di anni, l'upskilling non è un evento una tantum ma un processo strutturale, con aggiornamenti costanti su tecniche di prompting, governance degli output e metodologie di integrazione. **Terza**: l'innovazione, le risorse liberate dai task ripetitivi vengono reinvestite in R&D per sviluppare nuove offerte scalabili e sperimentare modelli di delivery AI-native.

# Il modello di adozione: come fare la differenza

Il report DORA (DevOps Research and Assessment) 2025 ha individuato un meccanismo critico: l'AI agisce come "mirror and multiplier". Nelle organizzazioni con solide pratiche ingegneristiche, l'AI amplifica l'efficienza. Nelle organizzazioni frammentate, amplifica le debolezze.

Questo spiega perché il progetto Assist Digital ha strutturato l'adozione con un approccio rigoroso: baseline KPI pre-AI, scaling graduale e validato da dati reali, evangelist di area come agenti del cambiamento, formazione calibrata sui feedback del pilota e dashboard ROI continuativa. Non il tool, ma il processo di adozione determina il risultato.

# Conclusioni: un’opportunità strategica da non sprecare

L’AI nel ciclo di sviluppo software è già realtà, non prospettiva. Il 2026 è l’anno in cui la trasformazione sistemica diventa evidente: non si tratta più di singoli developer più veloci ma di interi processi ridisegnati, team che cambiano forma, organizzazioni che competono su capacità di adozione oltre che su competenze tecniche tradizionali.

Per le aziende di IT services come Assist Digital, la posta in gioco è doppia. Da un lato, l’AI è un potente strumento di efficienza interna, come dimostra il business case del progetto della unit CX Technology. Dall’altro, è un’opportunità di riposizionamento competitivo: i clienti si aspettano sempre più che i partner tecnologici portino AI-native delivery, non solo persone che usano AI.

Le organizzazioni che avranno successo non saranno quelle che resistono al cambiamento né quelle che adottano ogni tool senza metodo. Saranno quelle che, come Assist Digital, costruiscono un modello strutturato: obiettivi chiari, dati reali, formazione specifica, governance dell’AI output e capacità di iterare velocemente sulle evidenze.

## La tesi di fondo

L’AI non sta rendendo obsoleti gli sviluppatori: li sta liberando dal codice meccanico per elevarli a un ruolo più strategico. Chi abbraccia questa evoluzione con metodo, costruendo AI literacy, ridisegnando i processi e governando la qualità degli output, diventerà più capace e più competitivo. Chi aspetta, si troverà a rincorrere un gap che cresce ogni mese.

# Appendice – Fonti e riferimenti


**(1)** McKinsey — 2025 State of AI Report: dati sull'utilizzo AI nelle funzioni aziendali

**(1)** **(9)** Stack Overflow — 2025 Developer Survey: dati su adozione tool AI e impatto su giovani developer

**(2)** **(5)** Anthropic — 2026 Agentic Coding Trends Report: analisi dell'evoluzione dell'agentic coding e impatto sul SDLC

**(2)** Second Talent — AI in Software Development Statistics 2026: raccolta di statistiche da fonti primarie su adozione, produttività e tendenze

**(3)** Anthropic Research — How AI Is Transforming Work at Anthropic (Dicembre 2025): studio su 132 ingegneri e 200.000 transcript di Claude Code

**(4)** DevOps Research and Assessment

**(6)** Assist Digital — Piano Strategico AI Adoption CX Technology (Marzo 2026)

**(7)** AI In Software Development di Grand View Research

**(8)** Index.dev – Top 100 AI Pair Programming Statistics 2026; 50+ AI Tool ROI Statistics

**(10)** Gloat — AI Workforce Trends 2026: analisi Gartner e PwC sull'impatto organizzativo

**(11)** SignalFire — State of Tech Talent Report 2025

**(12)** Stanford University — Canaries in the Coal Mine? Six Facts about the Recent Employment Effects of Artificial Intelligence

**(13)** Gartner, Keyhole Software, Hostinger — Ulteriori dati di mercato su sviluppo software e tendenze tecnologiche 2026

# Grazie

contact@assistdigital.com
www.assistdigital.com