TECHNOLOGY & INNOVATION

Artificial Intelligence in the Software Development Lifecycle

ASSIST DIGITAL

Executive Summary

IT services companies operate under increasing pressure: to innovate faster, ensure software quality and security at scale, and manage ever tighter margins. To address these challenges, the adoption of Artificial Intelligence within the Software Development Lifecycle (SDLC) is no longer an exploratory option but a strategic necessity. In 2026, approximately 80%[1] of developers use AI tools in at least one phase of the software lifecycle, and 40%[2] of global code is already generated or co-generated by AI. However, most organizations remain in a phase of partial adoption, focused on isolated use cases rather than on systemic transformation.

This document is the result of the synergy between the direct experience gained through the AI Adoption project within Assist Digital's CX Technology unit and an in-depth analysis of third-party reports and market studies. The document examines the impact of Artificial Intelligence on processes, technologies, benefits, and risks, providing an overall view of the organizational evolution required to govern this transformation.

Key Findings

AI is reshaping the entire SDLC, not just coding. he impact is not limited to code generation: from semantic requirements analysis to architectural design and testing, every phase is undergoing a qualitative transformation. The distinction between sequential phases is fading in favor of a fluid agentic workflow, with completion times shrinking from days to hours.

Productivity gains are real but require an effective governance and control model. Aggregated data indicates savings of between 30% and 60%[8] across coding, testing, and documentation. However, 62%[9] of developers report technical debt risks. Assist Digital has mitigated this "GenAI Paradox" by integrating AI directly into CI/CD pipelines with automated quality gates and a "human-owned" culture in which accountability remains human.

Agentic coding is the real turning point. The shift from simple autocomplete assistants to agentic systems capable of managing multi-step tasks in a semi-autonomous/autonomous manner is the most disruptive development of 2025–2026. Developers use AI tools in approximately 60%[5] of their activities, although full delegation remains limited to a small number of tasks.

The organizational impact is profound and cannot be postponed. AI does not eliminate technical roles but transforms them: developers evolve toward orchestration and validation, while senior developers move toward architecture and AI governance. The dynamic staffing model based on technology pools adopted by Assist Digital optimizes utilization by managing more services with the same AI-enhanced team.

It is not the tool, but the adoption process, that determines the outcome. AI acts as a "mirror and multiplier": it amplifies efficiency where solid practices exist and amplifies weaknesses where methodology is lacking. Assist Digital structured its adoption journey through pre-AI KPI baselines, gradual scaling, area evangelists, and continuous ROI dashboards, demonstrating that competitive advantage stems from process discipline, not from the power of any individual tool.

index

01 How AI is changing software development processes

02 The software development lifecycle phases: toward the multi-agent ecosystem

03 The most widely used AI technologies and emerging trends

04 Strategic benefits and mitigation of the “GenAI Paradox”

05 Organizational impact and dynamic staffing

The adoption of Artificial Intelligence in software development is no longer a future trend: it is a transformation already underway.

In 2026, approximately 80%[1] of developers use AI tools in at least one phase of the software development lifecycle, and around 40%[2] of global code is already generated or co-generated by AI.

This document analyzes the impact of this revolution on processes, technologies, benefits, risks, and organizational structures, drawing on the experience gained through Assist Digital's internal AI Adoption project for the CX Technology team.

TECHNOLOGY & INNOVATION

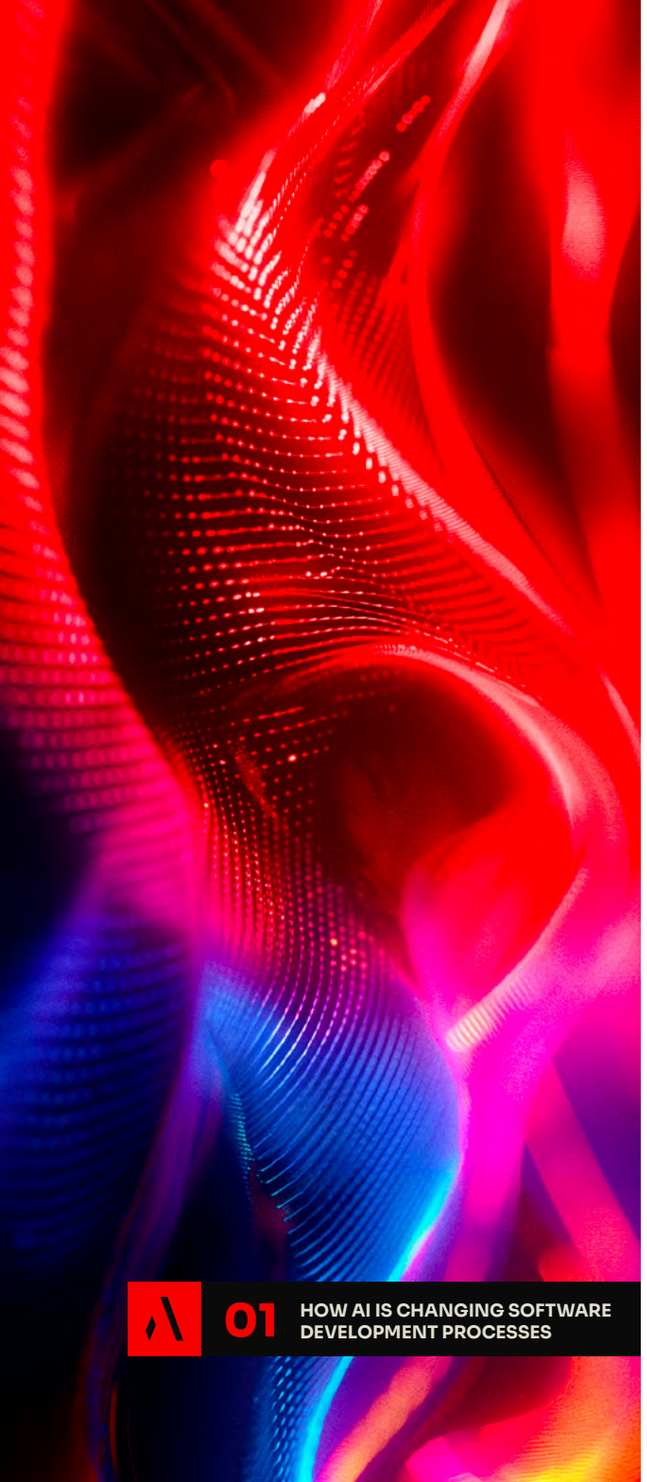
01

How AI Is Changing Software Development Processes

The software development industry is undergoing the fastest transformation in its history.

The adoption of AI tools within the developer workspace has surpassed even the transition to cloud computing and the introduction of Git in terms of adoption speed. In just three years, we have moved from a handful of experimental tools to a mature ecosystem that spans every phase of the Software Development Lifecycle (SDLC).

ASSIST DIGITAL



The turning point: from assistance to autonomy

Until 2023, the use of AI in the software development lifecycle was essentially limited to advanced autocomplete functionality. Today, we are witnessing a qualitative transformation: Large Language Models (LLMs) have evolved into agentic systems capable of understanding requirements expressed in natural language, generating architectures, writing tests, performing code reviews, and managing production incidents with reduced human supervision.

The concept of "agentic coding," namely the ability of AI tools to execute complex tasks in a semi-autonomous/autonomous manner by iterating on feedback and linking multiple steps in sequence, is the most disruptive innovation of 2025–2026. According to Anthropic's "2026 Agentic Coding Trends" report, we are witnessing a systemic redesign of the entire SDLC, with activities that previously required weeks of coordination among teams being transformed into focused work sessions.

SDLC phases: before and after AI

The impact of AI is not uniform across the entire software lifecycle. Some phases are undergoing profound transformation, while others are evolving more gradually. Generally speaking, the more codified, repetitive, and pattern-based activities are those in which AI demonstrates its greatest potential; activities that require contextual judgment, business understanding, and creativity remain heavily dependent on human oversight, at least in the short term.

The narrative of developer "replacement" has given way to a more nuanced and realistic perspective: AI as a capability multiplier, not a substitute. Google CEO Sundar Pichai summarized this view effectively: 25% of Google's code is already AI-assisted, but the metric that matters is the increase in overall engineering velocity, estimated at around 10%, not a reduction in the number of developers.

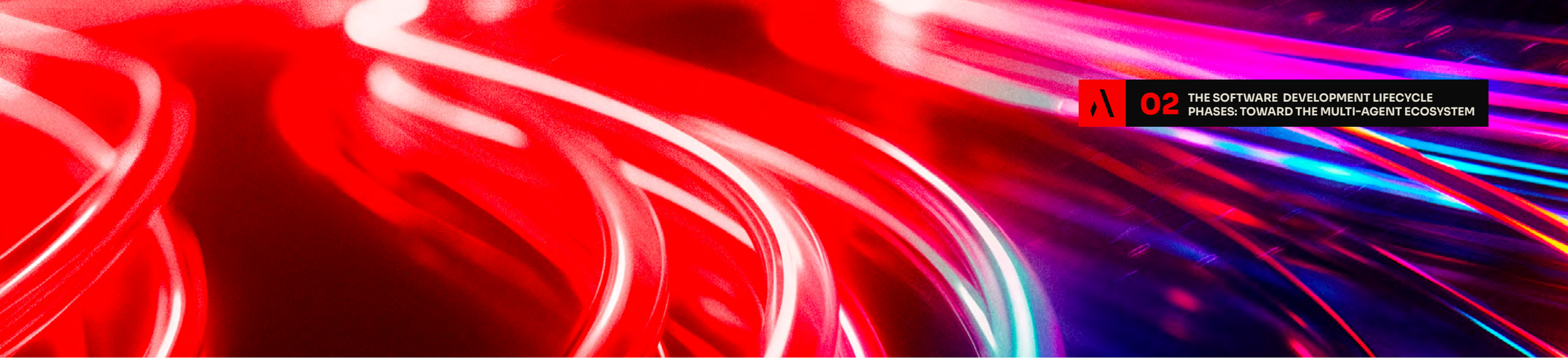
TECHNOLOGY & INNOVATION

02

The software development lifecycle phases: toward the multi-agent ecosystem

The impact of AI on the development lifecycle goes beyond isolated gains in speed: it redesigns the delivery model. The phases of the SDLC, historically sequential, are converging into a more fluid agentic workflow, in which tasks that once required days or weeks can be completed within a matter of hours.

ASSIST DIGITAL



Analysis and design: from collection to generation

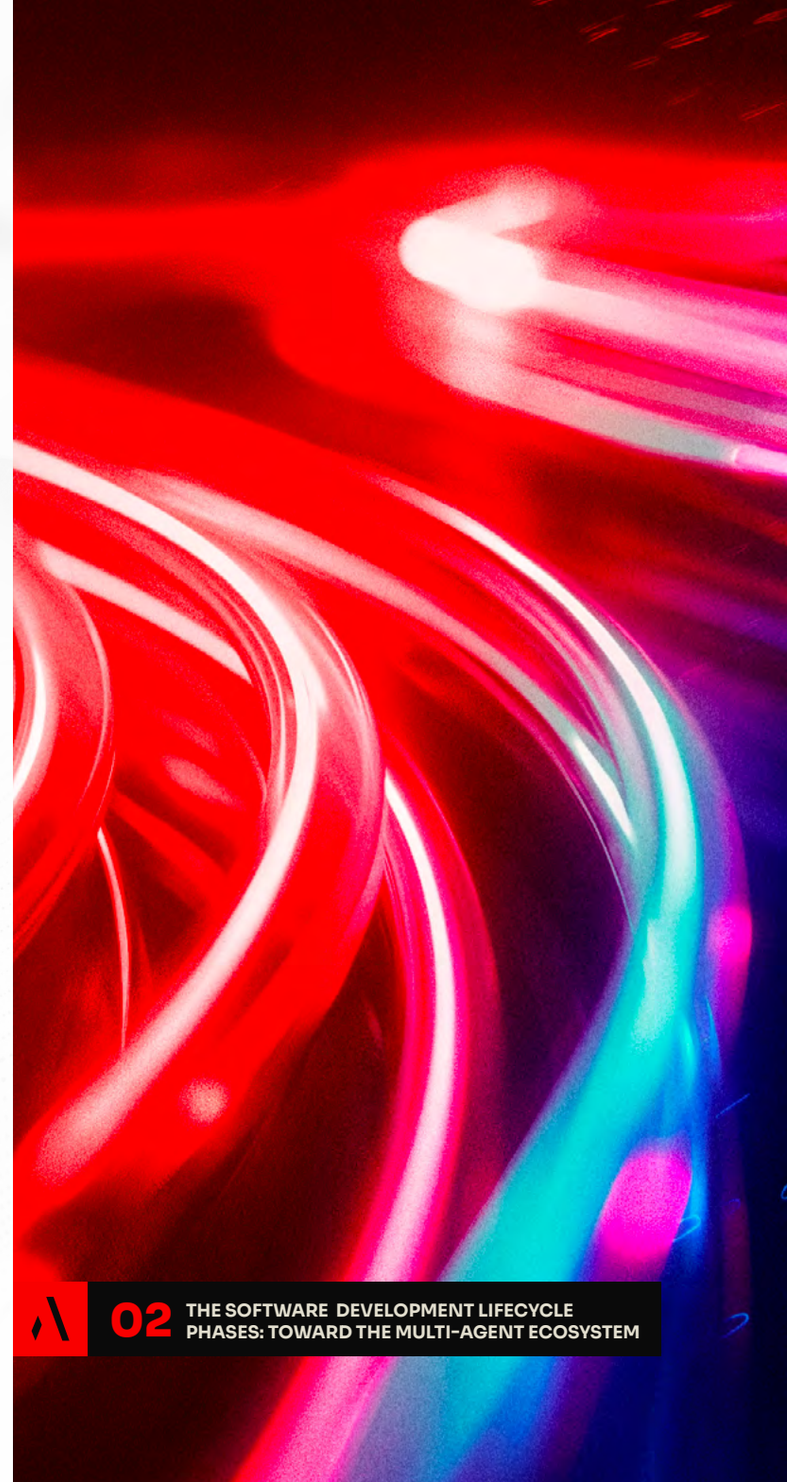
The adoption of agentic coding tools has transformed functional analysis from a labour-intensive activity into a process of intelligent synthesis:

- Semantic analysis: the automatic extraction of structured requirements from transcripts and informal documents, reducing ambiguities before development begins.
- Productivity increase: a perceived increase of between 20% and 30%[3] in the production of user stories and epics.
- Architectural design: Artificial Intelligence acts as an effectiveness multiplier for architects[4], supporting the definition of technology stacks and the simulation of scalability through mature Platform Engineering practices.
  This enables senior professionals to evolve toward roles focused on strategic oversight and architectural governance.

Software Development (Coding)

The coding phase is the most heavily impacted, with 40%[(2)] of global code being co-generated by AI. It is the phase most affected in absolute terms and the one that has received the greatest media attention. AI in coding manifests itself at multiple levels:

- Advanced autocompletion and code generation: from simple snippets to entire functions or modules.
- Intelligent debugging: identification of bugs during the coding process, with suggested fixes.
- Automated code review: pull request reviews focused on style, security, and compliance with standards.
- Refactoring and optimization: suggestions for code rewrites to improve performance and maintainability.
- Agentic coding: autonomous execution of multi-step tasks, from ticket analysis to code writing, testing, and documentation.



Software Development (Coding)

The most telling statistic: according to Anthropic's "2026 Agentic Coding Trends" report[5], developers use AI tools in approximately 60% of their activities, but only a limited share of tasks, currently below approximately 20%, is fully delegated to agents.

Testing and Quality Assurance

Testing has traditionally been one of the most critical phases of the software development lifecycle. AI introduces a profound transformation through:

- Automated test case generation, including scenarios that would be difficult to envision manually.
- Predictive testing: identifying areas of code with the highest risk of regression in order to focus testing efforts where they are most needed.
- Creation and management of regression test suites and stress tests.
- Intelligent release management: optimal selection of deployment windows.

Within the Assist Digital project, the Quality Assurance team uses agentic coding systems for test authoring and execution, resulting in an impact on code coverage and an estimated 10–15%[6] reduction in testing time.

Operations, monitoring, and maintenance

AI applied to IT Operations activities, often referred to as AIOps, represents the most mature frontier after coding. AI systems applied to IT Operations enable:

- Real-time performance monitoring with anomaly detection and predictive alerts.
- Automated correlation of IT events to identify the root causes of incidents.
- Automated analysis of maintenance tickets: classification, prioritization, and solution recommendations.
- Support for corrective maintenance: log analysis, correlation with source code, and bug-fixing recommendations.

This last capability is particularly strategic for IT services companies such as Assist Digital: AI can analyze the reported issue, correlate it with code history and application logs, and propose potential causes and fixes to developers, significantly reducing MTTR (Mean Time to Repair).

03

The most widely used AI technologies and emerging trends

ASSIST DIGITAL

The current technology landscape

The market for AI-powered software development tools is growing rapidly: from an estimated value of $933 million in 2025, it is expected to reach $15.7 billion by 2033, with a CAGR of 42.3%[7].

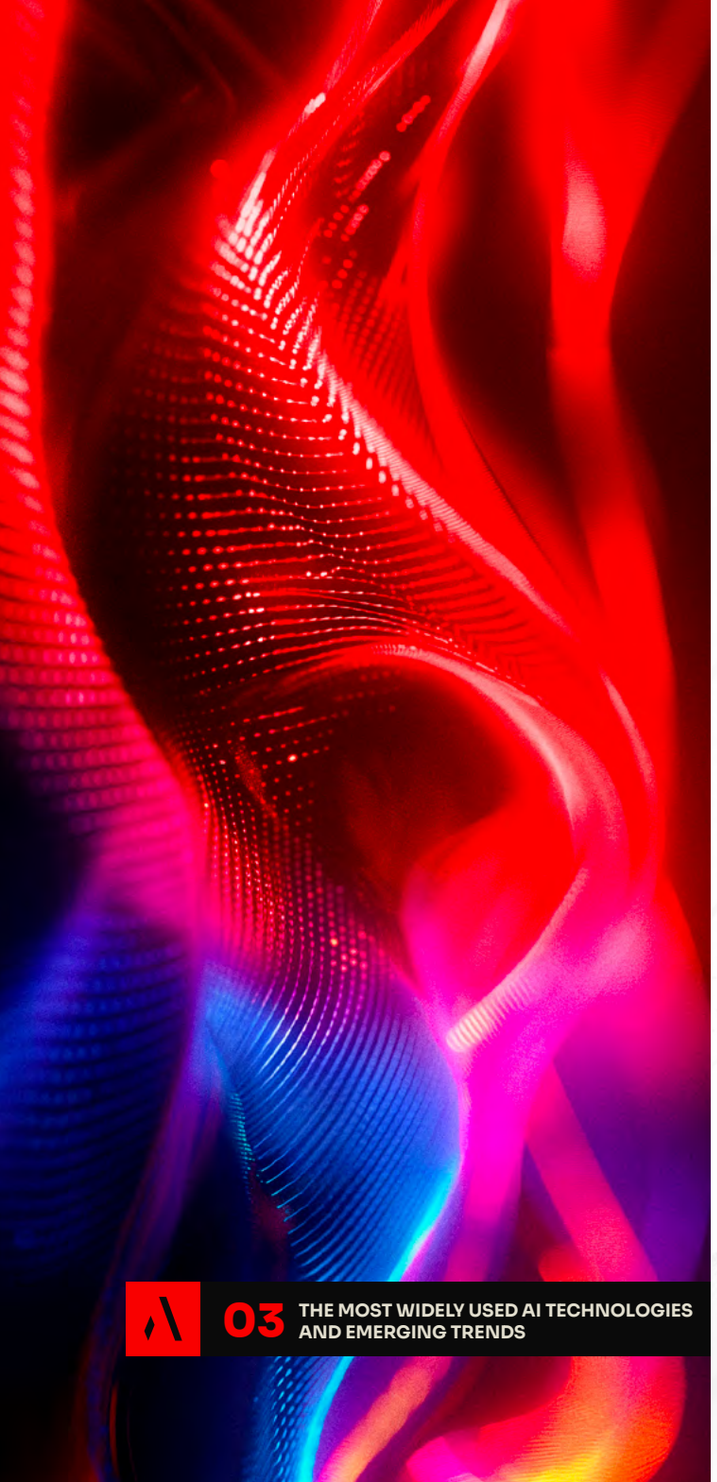
15,7 bln

AI market growth for software development by 2023

Frontier Large Language Models (LLMs)

Claude (Anthropic), GPT (OpenAI), and Gemini (Google) represent the three leading players, with model families designed for different use cases, from highly complex multi-step reasoning (e.g., Claude Opus, GPT-5.5) to real-time integration within development environments. The choice of model depends on specific characteristics: multi-step reasoning capabilities, agentic capabilities, tool use, context window size, response speed, and cost per token.

Within the Assist Digital project, the selected approach was a mix of coding agents as the primary tool for complex cognitive activities (coding, analysis, architecture, and testing), due to their excellence in multi-step reasoning, agentic capabilities, and code generation quality, combined with conversational GenAI tools already available within the corporate IT ecosystem for everyday general-purpose activities.



Coding Assistants and IDEs

GitHub Copilot and Claude Code are the tools emerging with the highest adoption rates, but the market has become increasingly fragmented with the rise of specialized solutions. The dominant trend is the direct integration of these tools into IDEs (Integrated Development Environments): they are no longer standalone applications but integrated copilots embedded within the workflow, with access to the full codebase context.

Workflow orchestration and automation

The second frontier is process automation through AI workflow orchestrators. One example is n8n, an orchestration platform that enables the creation of trigger-based workflows with on-demand AI capabilities, integrated with existing enterprise systems (i.e., CRM, Ticketing, Monitoring, CI/CD, and Cloud environments).

Emerging Trends: Agentic AI and Multi-Agent Systems

The most significant trend for 2026 is the shift from single-agent systems to multi-agent architectures. In these systems, specialized agents cooperate: one analyzes the requirements, a second generates the code, a third runs the tests, and a fourth generates documentation, all in a coordinated and semi-autonomous/autonomous manner.

The practical result is already visible: Amazon has estimated savings of 4,500 developer-years and $260 million in a single migration project by using AI coding integrated into the organizational infrastructure, rather than as an individual tool.

Assist Digital's technology framework

The AI Adoption project within CX Technology has organized AI tools across four levels, ranging from the most sophisticated to the most immediate. Each level addresses different needs, but they work together.

At the first level are agentic coding tools: development environments in which AI agents autonomously perform entire tasks, from the analysis of a request through to code writing, testing, and documentation, always under human supervision. Two elements make these tools truly powerful. The first is Skills: reusable instructions that encapsulate the best practices, technical choices, and standards of our projects, so that the agent knows "how we work here." The second is MCP (Model Context Protocol) connectors, which allow the agent to interact in an orderly way with our systems and with external systems: code repositories, cloud environments, ticketing systems, monitoring, and documentation. Together, shared Skills and MCP connectors transform agentic coding from a personal tool into a shared resource for the entire team.

At the second level is a conversational GenAI assistant, used every day by everyone for common activities: summarizing documents, brainstorming, writing texts, and analyzing information on the fly.

At the third level is a workflow automation platform, which makes it possible to create automated processes triggered by events, with AI stepping in when needed, and connected to the systems we already use (Service Management, reporting, ticket classification).

Finally, at the fourth level, language models and agentic systems are integrated directly into development environments. In this way, AI is not a separate application, but a natural component of engineers' daily work.

This multi-level structure allows us to balance power, cost, and data control, without tying ourselves to a single provider.


TECHNOLOGY & INNOVATION
04
Strategic benefits and mitigation of the “GenAI Paradox”
ASSIST
DIGITAL

A value-oriented business case

Productivity and delivery speed

The impact on productivity is the most frequently discussed outcome. Aggregated data indicates an increase of between 30% and 60%[8] in time saved across coding, testing, and documentation. This perception is also reflected in our direct experience: when the results are compared with feedback from the managers of the various development areas, delivery time is almost never the first benefit they mention. What emerges most strongly is the ability to make better, more informed, faster, and more data-driven decisions, as well as improved collaboration among teams that previously operated in silos. Individual developer productivity is acknowledged, but almost as a natural consequence of a different way of working rather than as the starting point. This is an important signal: the value of AI within the SDLC does not end with gains in individual productivity, and the organizations that understand this first are the ones that derive the most enduring benefits from it.

Code quality and defect reduction

With AI suggesting corrections in real time during development, bugs are identified before they reach the repository. Automated pull request analysis reduces the workload required for manual code review while simultaneously expanding the coverage of quality checks. The expected result is a reduction in the number of production defects and improved test coverage.

Democratization of skills

AI enables developers to operate beyond their own area of specialization: a backend developer can tackle frontend tasks with AI support, a functional analyst can produce high-quality technical documentation, and a service management operator can automate complex workflows without programming skills. This "full-stack democratization" effect increases organizational flexibility.

30%

TIME SAVED ON CODING WITH AI TOOLS[8]

60%

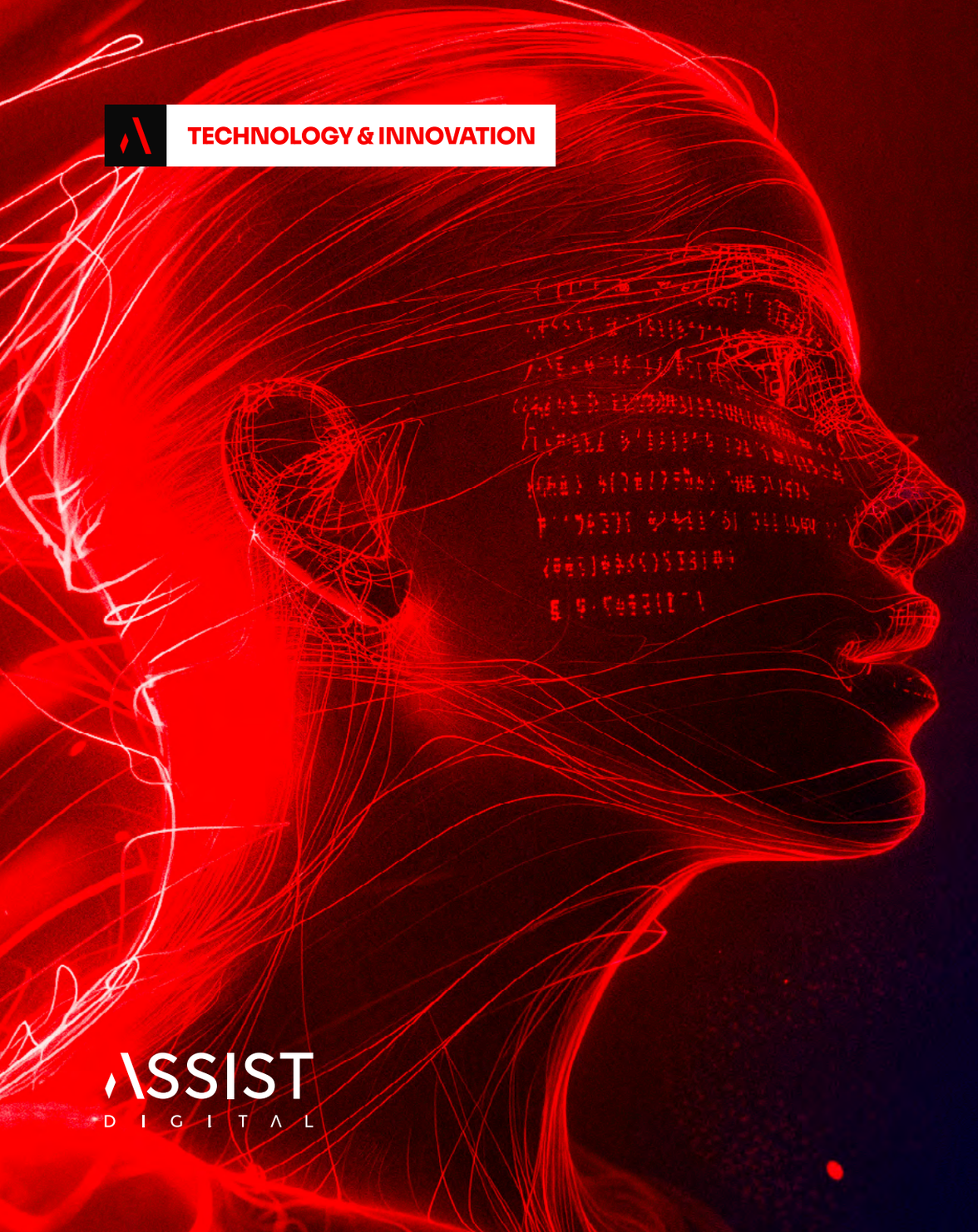
Addressing critical issues: Assist Digital's approach

The adoption of AI in software development brings with it a concrete paradox: code production accelerates, but without the right safeguards, technical debt can accumulate at the same pace. More generated code means more surface area to maintain, more potential vulnerabilities, and more architectural inconsistencies. To mitigate this risk, Assist Digital has structured its adoption around three complementary levers.

- **The first is the systematic strengthening of continuous integration and continuous delivery processes.** AI-generated code is subjected to the same checks — and in some cases additional checks — as manually written code: automated quality controls, static code analysis, security checks, and test coverage levels higher than the thresholds in place before the introduction of AI. The principle is simple: if AI produces more code, more controls are needed, not fewer. Although 62%[9] of developers report technical debt risks associated with the use of AI, integrating these safeguards directly into the release flow, rather than treating them as subsequent activities, makes it possible to identify and correct critical issues before the code is integrated into the repository.

- **The second lever is the role of senior profiles.** Experienced developers and Architects become the supervisors of AI-generated artifacts: they do not write less code because AI writes it for them, but rather invest the time freed up in critical review, the continuous updating of AI usage best practices, and the strengthening of project methodologies. This is a transition from "code writer" to "guarantor of architectural consistency and overall system quality." Without this oversight, the risk is that AI will produce solutions that are locally correct but globally inconsistent, insecure, and even less maintainable.

- **The third is a human-owned culture.** Responsibility for what goes into production always remains human, and for this reason the decision must remain human as well. A person cannot be accountable for a choice they did not truly make. Skepticism toward what AI produces is not discouraged, but valued as a natural control mechanism. AI is a powerful assistant, not an autonomous decision-maker, and the organization is structured to ensure that no one treats it as one.



TECHNOLOGY & INNOVATION

05

Organizational impact and dynamic staffing

ASSIST DIGITAL

The evolution of roles

AI does not eliminate technical roles but radically transforms them, elevating every profile toward higher-value activities.

- Role evolution: developers focus on validation and tool orchestration; senior developers and architects become the guarantors of architectural consistency and security oversight.
- Technology pools: optimization of utilization, as the same team becomes more versatile and capable of working across technology stacks that would previously have required separate skill sets.
- Continuous innovation: resources freed from repetitive tasks are reinvested in R&D activities aimed at developing new scalable offerings.

Emerging skills

An increasing number of companies are investing in AI, yet many report a skills shortage that is slowing adoption. Recent reports highlight that, as AI automates repetitive tasks, human skills such as critical thinking, communication, leadership, and business understanding become increasingly important.

AI literacy: understanding how models work, their limitations, effective prompting techniques, and best practices for supervising outputs.

Agentic workflow design: the ability to design automation pipelines that combine AI, enterprise systems, and human intervention at the appropriate points.

Quality & Security oversight: the skills required to critically evaluate AI-generated code, identify security vulnerabilities, and manage AI-induced technical debt.

The impact on the labor market is already visible: PwC reports that professionals with advanced AI skills earn up to 56%[10] more than peers in the same role who do not possess such skills.

The Organizational Pyramid

The impact on organizational structure is one of the most debated and sensitive aspects. Two opposing dynamics are taking place simultaneously:

The shrinking base of the organizational pyramid

The activities traditionally associated with junior profiles (standardized coding, simple debugging, manual testing, and documentation) are the most automatable. The data confirms this: hiring of entry-level technology professionals declined by 25% year-over-year in 2024[11]. A Stanford study highlights a 20%[12] decline in employment among software developers aged 22 to 25 compared to the 2022 peak. More than 50%[13] of senior developers anticipate a reduction in entry-level tasks, resulting in leaner and more specialized teams.

Expansion of the middle layer and governance

Conversely, roles focused on oversight, architecture, and AI governance are growing. New positions are emerging, such as AI Engineer, AI Architect focused on AI infrastructure, and Security AI Specialist. Gartner predicts that AI will create entirely new categories of roles in software engineering and operations that did not exist five years ago.

Assist Digital's approach: dynamic staffing through technology pools

First, the redesign of roles. Developers evolve from code writers into orchestrators of AI tools and validators of outputs, while senior professionals take responsibility for architectural consistency, security oversight, and the supervision of multi-agent systems. **Second**, continuous training. In a context where tools, models, and best practices for AI applied to the SDLC evolve in cycles measured in weeks rather than years, upskilling is not a one-time event but a structural process, with continuous updates on prompting techniques, output governance, and integration methodologies. **Third**, ongoing innovation. Resources freed from repetitive tasks are reinvested in R&D to develop new scalable offerings and experiment with AI-native delivery models.

The adoption model: how to make a difference

The 2025 DORA (DevOps Research and Assessment) report identified a critical mechanism: AI acts as a “mirror and multiplier.” In organizations with strong engineering practices, AI amplifies efficiency. In fragmented organizations, it amplifies weaknesses.

This explains why the Assist Digital project structured its adoption journey around a rigorous approach: pre-AI KPI baselines, gradual scaling validated through real-world data, area evangelists acting as change agents, training calibrated on pilot feedback, and continuous ROI dashboards. It is not the tool, but the adoption process, that determines the outcome.

Conclusions: a strategic opportunity not to be missed

AI in the software development lifecycle is already a reality rather than a future prospect. 2026 is the year in which systemic transformation becomes evident: it is no longer a matter of individual developers working faster, but of entire processes being redesigned, teams changing shape, and organizations competing on their ability to adopt new technologies as much as on their traditional technical expertise.

For IT services companies such as Assist Digital, the stakes are twofold. On the one hand, AI is a powerful tool for internal efficiency, as demonstrated by the business case of the CX Technology unit's project. On the other hand, it represents an opportunity for competitive repositioning: clients increasingly expect technology partners to deliver AI-native solutions, not simply people who use AI.

The organizations that will succeed will not be those that resist change, nor those that adopt every tool without a method. They will be those that, like Assist Digital, build a structured model: clear objectives, real-world data, targeted training, governance of AI-generated outputs, and the ability to iterate rapidly based on evidence.

Core thesis

AI is not making developers obsolete: it is freeing them from mechanical coding tasks and enabling them to take on a more strategic role. Those who embrace this evolution methodically, by building AI literacy, redesigning processes, and governing the quality of outputs, will become more capable and more competitive. Those who wait will find themselves chasing a gap that grows wider every month.

Appendix — Sources and References

(1) McKinsey — 2025 State of AI Report: data on AI adoption across business functions

(1) (9) Stack Overflow — 2025 Developer Survey: data on AI tool adoption and its impact on junior developers

(2) (5) Anthropic — 2026 Agentic Coding Trends Report: analysis of the evolution of agentic coding and its impact on the SDLC

(2) Second Talent — AI in Software Development Statistics 2026: compilation of statistics from primary sources on adoption, productivity, and trends

(3) Anthropic Research — How AI Is Transforming Work at Anthropic (December 2025): study involving 132 engineers and 200,000 Claude Code transcripts

(4) DevOps Research and Assessment

(6) Assist Digital — CX Technology AI Adoption Strategic Plan (March 2026)

(7) AI in Software Development by Grand View Research

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

(10) Gloat — AI Workforce Trends 2026: Gartner and PwC analysis of organizational impact

(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 — Additional market data on software development and technology trends in 2026

Thank you

contact@assistdigital.com
www.assistdigital.com