ESP-JETA
ESP Journal of Engineering & Technology Advancements
ISSN: 2583-2646 / Volume 6 Issue 2 April to June 2026
Paper Id: JETA-V6I2P105/ Page No: 37-46

Original Atricle

Preparing the Enterprise for AI-Driven Software Development: A Readiness Framework for Organizational Transformation

Debashis Patra[1], Ambar Nath Saha[2]
[1]*AI, UT Austin, Austin,USA*
[2]*Engineering Manager, Cognizant Technology Solutions Canada Inc, Canada*

Received Date: 11 March 2026 Revised Date: 20 March 2026 Accepted Date: 08 April 2026

Abstract: *The whole human race is scurrying towards introducing the Artificial Intelligence in their work streams and the software development companies are on the frontline. However, in practice, the majority of software development organizations are finding it challenging to get beyond small AI experiments(1), and they are finding themselves using AI as a code assistant or business advisor. The cause is not on the technology itself but organizational issues like skills deficiency, ineffective processes, poor governance and resistance to change. In this paper, we put forward the Enterprise AI Coding Readiness Framework (EACRF), which assists companies to determine their readiness to embrace AI in their software development lifecycle. The framework primarily examines five crucial dimensions namely Infrastructure, Skills, Processes, Governance as well as Culture. It also outlines a realistic three-step adoption process, which begins with AI-assisted development, proceeds to AI-enhanced processes, and ultimately to AI-first model. We also present a six-layer security solution to stop sensitive information leaks, a centralized knowledge system with intelligent context layer to give superior inputs to AI, a structured learning process to enable employees to adjust, and a cost optimization plan by using multiple AI models with a routing scheme according to the complexity of the task. We use a financial services case to illustrate how this framework can be used to address critical skill and governance gaps in a systematic manner within a finite time. The framework also provides ways in which organizations can proceed with a series of work streams running concurrently in a manner that, infrastructure, skills, governance and other aspects enhance each other rather than individually. This work aims to offer engineering leaders a transparent and practical direction in order to assess readiness, focus investments, and transition to large-scale AI usage in a controlled and sustainable way(2).*

Keywords: *Enterprise AI readiness, AI-driven software development, organizational transformation, AI governance and security, multi-model AI strategy.*

I. RELATED WORK

A. Adoption of technology and AI in businesses

Diffusion of innovations (4) by Rogers and the Technology-Organization-Environment framework (5) created a baseline in the adoption of technology in enterprises. Ransbotham . (6) discovered that the disjunction between AI ambition and implementation is largely organizational. According to Fountaine (7), scaling AI has to be based on a paradigm shift in its operating model. Nonetheless, these articles cover AI adoption in general, but not AI-based software development(1) in particular..

B. Developer Experience and AI Safety

Vaithilingam (8) discovered that developers were not finding it easy to have the calibration of trust when AI coding tools are used.Ziegler (9) came out with the fact that timely engineering and output appraisal expertise is necessitated. Ziegler (9) identified that effective use requires prompt engineering and output evaluation skills. On the security front, the Open Worldwide Application Security Project (OWASP) Top 10 for LLM Applications (10) list includes threats such as prompt injection, data leakage, and insecure output handling, which are all directly applicable to the use of enterprise AI code. The cultural dimension is informed by Kotter change model (11) and research by Westerman et al. on digital transformation (12).

II. ENTERPRISE A.I. CODING READINESS FRAMEWORK

The EACRF is composed of five readiness dimensions that are interdependent and each assessed on a scale of maturity ranging between Level 1(Initial) and Level 5 (Optimized). These two dimensions are very interrelated and cannot be separated (3). As an example, a robust infrastructure in the absence of appropriate skills can result in underutilization of the tools whereas

lack of appropriate skills in the absence of a proper governance would be a risk. On the same note, a culturally misaligned governance can bring resistant forces and delay the implementation.

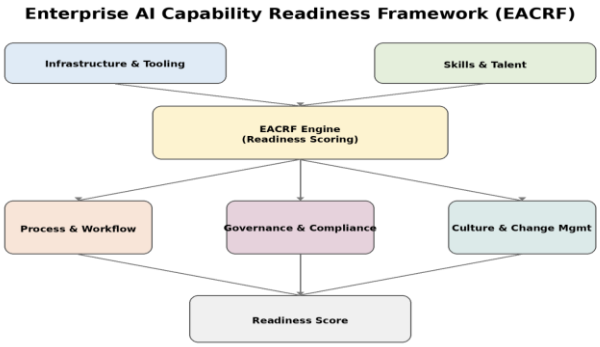

Figure 1: Enterprise AI Coding Readiness Framework — Five Interdependent Dimensions with Maturity Scale

A. Infrastructure and Tooling

It is also necessary to consider the basic set-up that organizations already have before beginning to use AI in the development of software. In most cases, organizations attempt to apply AI over the already in place legacy systems, which are not prepared to interface with AI and this is where the issue begins. The AI tool cannot provide the best results because the company should have a strong foundation to be integrated with the AI.

An extensive development setting is needed. This involves good source management, current development tools, and a fluent method to tie AI into daily workflows (6) in such a way that it is an extension of how developers work and not something distinct that they will need to maintain off the record.. In case the set-up is scattered or not user-friendly, teams might not use AI at all.

The other key point is the integration and access to AI systems. Organizations should make sure that AI model access is stable, secure, and in line with internal policies. It involves awareness of the destination of the data, data processing, and compliance. This clarity is missing and teams might be reluctant to utilize AI even when it exists.

Automation is also important. In case the build, testing, and deployment processes remain slow or rely on manual processes, the introduction of AI can only make the processes more complex rather than efficient. Quick and efficient CI/CD pipelines can assist teams to confirm AI-generated modifications within a short period of time (6) and minimise the chances of making errors..

In essence, AI should be facilitated by infrastructure in a manner that is smooth and trustworthy. AI can bring real value in case the foundation is strong. However, when the ground is not strong, it may cause confusion and slowness rather than assisting.

B. Skills and Talent Transformation

The advent of AI alters the position of developers in a major aspect. Developers are currently required to lead AI systems rather than write code, analyze their outputs, and make decisions based on the output of AI. Code writing is no longer a core competency of a developer, but progressively more time is spent by a developer in configuring MCP, model determination and configuring agents according to their applications. The new skills needed during this shift include the skill to pose the correct questions, analyze the responses generated by AI and knowing when to trust the result and when to doubt.

This process does not come automatically and needs the right learning support in the organization. Practical experience can often be more significant than theory because in most instances, teams have to apply AI to real-life situations to develop confidence and knowledge. With time, organizations can also begin to experience the birth of new positions that are aimed at enhancing the application of AI in the development processes. And even the most advanced tools can be of no use without this investment in skills.

C. Process and Workflow Adaptation.

The current software development processes were developed under the assumption that most of the work is done by humans. These processes should be modified with the introduction of AI into the workflow. Indicatively, it is no longer possible

to do small code reviews since AI can produce large bodies of code simultaneously. The reviewers should be more concerned with logic, accuracy and design in general, and not just syntax.

Testing also becomes more imperative. Although AI can be used to create test cases, the test cases have to be validated to provide meaningful and effective tests. There is a risk of blindly relying on AI-generated tests. Moreover, the teams can gradually transition to less fixated sprint-based models to more continuous processes where the development and validation are more frequent.

Documentation also acquires a different level of significance. Clear and well-organized documentation gives AI systems a good context to enhance the quality of their outputs. In this regard, documentation is no longer only to serve the human understanding but turns out to be a significant input to the whole development process.

D. Governance and Compliance.

The use of AI introduces a new type of issue in relation to other traditional systems, particularly in terms of data, ownership, and compliance. The reason why many organizations end up being vigilant at this point is correct since it is not always obvious what they are sharing with the AI systems and how it is utilized.

Organizations should ensure that they are clear about the type of data that can be shared and what must be kept confidential at all times. External models should never be exposed to sensitive information like API keys, personal data, or internal business logic. Any little slip at this point may cause severe ramifications, and this is why groups must be cautious of the way AI is incorporated into their operations.

Ownership and accountability is another area which requires clarity. In case the AI produces code or any output, there must be an understanding of the responsible party. Concurrently, there are still some steps that are crucial and require a human touch, in particular, where there are compliance or regulations. This will assist in preventing the blind faith in AI and allow important decisions to remain under control.

It is also essential to have appropriate audit trails. Teams can be able to trace how a given output was produced, what model was employed and who was reviewing or approving it. This not only assists in compliance, but also instills confidence in the organization during AI use.

Unless these aspects of governance are managed appropriately, organizations can be put at risk of halting or even halting the adoption of AI. However, with visibility and clear rules in place, teams are much more comfortable and the adoption process is much easier.

E. Culture and Change Management

People are the real challenge, but all the technical and process-related changes are important. In many organizations, there is still a lot of uncertainty around AI. Others believe that it may take their job away, and others are yet to be convinced on the value of the same. Due to this, adoption does not occur naturally though the tools are available.

It is necessary to emphasize that AI does not come to eliminate people, but to transform the manner in which people work. As an example, developers might have less time to write everything and more time to review, reason about issues and make better choices. This change does not occur within a day, and it requires the assistance in the form of education, coaching, and motivation.

Here also leadership has a significant role to play. It is not sufficient to give access to AI tools. Teams must feel free to use them, to experiment with them and to make mistakes in the learning process. Concurrently, various teams including engineering, security, legal, and business must be heading in the right direction. Without alignment of one of the groups, the overall progress can be slowed down.

Finally, without the readiness or confidence of the people, it will be difficult to succeed even with the best structured system. Adoption is far more comfortable when the whole team is informed of the cultural change and embraces the change to their greater advantage.

III. MODEL SECURITY TESTING

The question of whether an organization will be safe with its data or not is one of the biggest concerns of any organization prior to using AI in the soft development process. Companies deal with a lot of sensitive data such as source code, API keys,

credentials (10), and customer information that cannot be exposed to the outside world. Due to this, the security becomes a necessity and not a luxury.

Our framework addresses this important aspect by providing a number of validation layers that will serve as gatekeepers. All the requests and responses go through multi-layered validation before interacting with any external AI systems. Prior to transmitting the data to AI models, our framework will ensure that the data is secure, pertinent, and consistent with internal policies (10) and compliance.

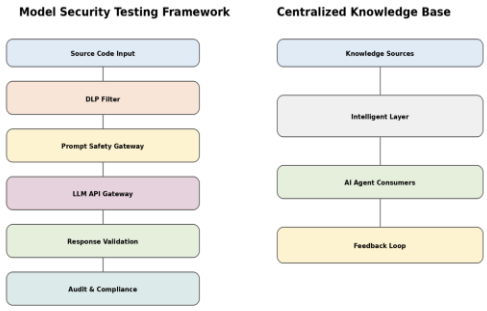

Figure 2: Model Security Testing Framework (left) and Centralized Knowledge Base Architecture (right)

Layer 1 (Source Code Input): The very first layer is the source code input where the code written by the developer is captured and the first thing that the system does is read the information and then know what information is being sent to be processed.

Layer 2 (DLP Filter): It is the layer that checks all the sensitive data including API keys, the private keys, credentials, internal hostnames and the business logic of the core business, etc. In case any such data is spotted, this layer tends to either eliminate it, or mask it and then forward the information to the subsequent layer.

Layer 3 (Prompt Safety Gateway): It verifies potential prompt injection attempts, constraining unnecessary context and only required information is passed to subsequent step. Meanwhile, the requests are all logged to enable tracking.

Layer 4 (LLM API Gateway): It is the layer that deals with routing of the request to approved AI models. This can be in the form of private endpoints, self-hosted models, or secured network channels depending on the organization settings. This is aimed at making sure that the systems in use are trusted.

layer 5 (Response Validation): This layer checks the response provided by the AI. It searches an inaccurate or illusive answer. It also verifies any feasible license problems or any sensitive information that might have been accidentally left out. Only proven outputs can be proceeded.

Layer 6 (Audit Trail): This layer maintains a record of all the interaction. This involves what data was transmitted, what model was applied and how the output was accepted. This aids in ensuring transparency, as well as compliance requirements.

This model is based on the zero-trust approach (15), implying that no data is regarded as safe in default. All inputs and outputs are checked prior to use or sharing. This will minimize the chances of data leakage (10) and create trust in the use of AI systems.

IV. CENTRALIZED KNOWLEDGE BASE - INTELLIGENT LAYER.

The performance of AI coding agents only depends on the context they are provided. Due to this fact, it becomes highly important to have a centralized knowledge system. The Centralized Knowledge Base is a common memory that can be accessed by various agents all through the lifecycle of software development and it is always changing. We can see the entire architecture as three components: knowledge sources, the intelligent layer and agent consumers.

The knowledge sources consist of various kinds of information that are already available in the organization like coding standards, API specifications, architecture documents, past test results, incident logs, compliance rules, reusable code patterns, team runbooks, and deployment configurations. This information is present and in most instances, it is dispersed across systems. The combination of these enables them to be inputs to the Intelligent Layer.

It is the Intelligent Layer that will make this information useful. It aids in searching the appropriate data either by semantic search and vector embeddings (2), and also ranks the data that is more helpful in a specific task. Besides this, it integrates data collected by different sources to create superior context to AI systems. It also gets better with time, which gathers the consequences of the past interactions, which aids in enhancing the quality of future responses.

This layer is interacted with by different agents like planning, architecture, code writing, testing and operations in order to obtain the information they require. Concurrently, they as well give back to the Knowledge Base according to what they produce or acquire in the course of performance. This forms a cyclic learning process (7) in which the system improves as time goes by.

In the long run, this strategy assists in creating superior and more cohesive results as the agents are not operating alone. They are led by precedents, familiar problems, and accepted tendencies in the organization. This is in some way what makes the system unlike traditional AI coding tools which do not have any long-term memory.

An illustration of how we can apply Modern large language models, like Claude, to agentic systems is to treat them as reasoning engines that act on structured prompts, produced by orchestrator agents. In this architecture, agents break down tasks and access contextual knowledge and dynamically forward requests to the right model levels (e.g., Claude Haiku, Sonnet, or Opus) depending on complexity, cost, and latency considerations. This allows context-aware and scalable implementation throughout software development lifecycle.

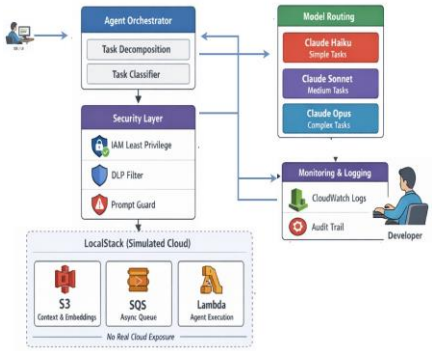

Figure 3: This Architecture Shows How Enterprise

AI agents may run in a controlled environment with LocalStack to emulate cloud services and provide strict least privilege access policies. IAM controls only allow each agent to access authorized resources, and DLP filtering, timely safety validation, and audit logging are used to guarantee safe and compliant implementation of AI-based workflows.

V. EMPLOYEE AI LEARNING PATH

The most weighted dimension in the EACRF is workforce skill transformation with a weight of 25% of the composite score. In most instances, it turns out to be the best indicator of whether the adoption of AI will turn out to be successful (6). It is because of this that the Employee AI Learning Path is meant to be a four-level progression which is compatible with organizational roles as well as with various phases of AI adoption.

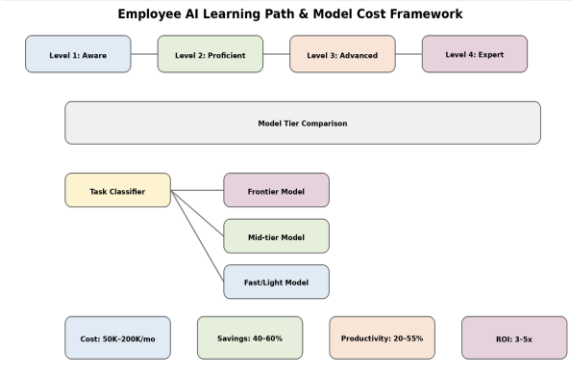

Figure 4: Employee AI Learning Path (top) and Model Cost Analysis with Selection Framework (bottom)

Level 1 (AI-Aware, Weeks 14): This is aimed at training the fundamental core of all engineering employees. All developers and engineers will begin to learn capabilities and limitations of LLMs, the general AI tools environment, basic data privacy terms and conditions, acceptable use policies, and their first practical AI-assisted task at this stage. The concept behind this is primarily to make people feel at ease with AI and what it can and cannot do.

Level 2 (AI-Proficient, Months 2-4): At this level, software and DevOps engineers will begin to apply AI in their everyday tasks. The techniques at this level are timely engineering, output assessment, trust calibration (8,9), AI-assisted code review, test generation, and pair programming with AI agents. Once this is done, all of them will be capable of implementing AI in their day to day duties.

Level 3 (AI-Advanced, Months 48): This level is more pertinent to tech leads, architects, and AI champions who are anticipated to drive AI adoption in teams. This tier includes agent workflow design, governance and compliance of AI, architecture with AI agents, and mentoring skills. Now, the learning is not merely learning how to use AI effectively but also learning how to make people use AI effectively.

Level 4 (AI-Expert, Months 8-12+): The last tier will be aimed at AI Engineering Leads and Prompt Engineers who assume more responsibility in terms of enterprise AI adoption. This tier is about multi-agent coordination, LLM guardrails and safety, model fine-tuning assessment, enterprise AI approach, and cross-organizational enablement. These people are also in most instances engaged in developing the manner in which AI is employed in the organization.

The enablement methods are meant to facilitate the whole learning process so the learning of skills is not restricted to theory. It involves practical workshops with real codebases, pairing AI-experienced and AI-new developers, curated internal prompt libraries, frequent retrospectives to reflect on what worked and what did not, and hackathons that enable a low-risk environment where experimentation is safe. In addition, competency assessments with certification are included at each level. All of the levels are certified to make sure that a certain standard of competence is attained before proceeding.

VI. MODEL COST ANALYSIS

One of the most important strategies of implementing Artificial Intelligence in organizations is financial planning. Costs of the LLM API may vary widely (2) based on the tier of model selected. No organization should adopt a single LLM to all the kinds of work that will result in either an unwarranted cost or a poor work. That is why, we have divided models into four levels and suggested a multi-layer routing policy to achieve the compromise between the cost and the quality of the output.

Frontier models (Claude Opus, GPT-40, Gemini Ultra) usually cost in the range of $15-17 per million input tokens, offer ultimate quality and can be applied to very complex architectural design and critical implementation of business logic etc.

Mid-range models (Claude Sonnet, GPT-40-mini) Price is between 3-15 and it offers an adequate compromise of price and performance. These Mid-tier models should be used in case of any standard coding, testing and code reviewing.

Simple tasks like documentation, boilerplate generation etc are normally performed on fast or lightweight models (Claude Haiku) that have a cost of about $0.253.

Besides the aforementioned levels, organizations can attempt to implement a pair of open-source self-hosted models (Llama, Mistral) but this comes at the expense of infrastructure.

Multi model routing strategy (7) should be adopted to ensure that the cost of the whole SDLC is not high with the output being of good quality. It will have a Task Classifier that will examine the complexity of each request and send it to the most suitable level of model. High complexity tasks like architectural design or novel algorithms should be taken care of by frontier models. Bug fixes can be sent to middle-level models, simpler tasks, such as boilerplate generation, documentation, are delegated to lightweight models. By doing this, an organization will be able to save about 40-60% of the costs it would have incurred when implementing only the high-end models.

The following is an approximate estimate of one of the enterprises that has approximately two thousand engineers, the monthly Api cost can be estimated between $50K and $200K. Yet it is extremely reliant on the effectiveness with which the organization selects model levels of different work. When you take the model selection and improvements in developer productivity of about 20-50% you can have a pay back of about 3-5x in the first year with the approach. The API cost can in most cases be offset by the less time of development and superior engineering throughput.

VII. THREE-PHASE ADOPTION STRATEGY

The EACRF suggests a three-phase adoption strategy so that organizations do not try to adopt AI all of a sudden. Early scaling that is done without adequate preparation produces problems (6) rather than opportunities in most occasions. then it is better to get slowly ahead and step by step.

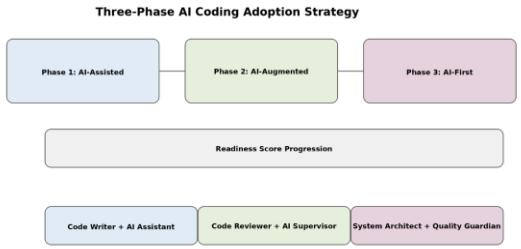

Figure 5: Three-Phase Adoption Strategy: AI-Assisted, AI-Augmented, and AI-First with Readiness Scores

A. Phase 1: AI-Assisted (Months 0–4).

This is where the developers begin using AI primarily to assist with code in their current workflow. Nothing significant does here change. The objective is to become acquainted with AI tools. To this, organizations are expected to already have in place basic things such as source control, CI/CD, and security checks. Participation can be optional in the beginning.

To proceed, visible daily usage of approximately 60 percent should be apparent, as well as consistent quality of code, simple policies, and certain degree of leadership support.

B. Phase 2: AI-Augmented (Months 4–10).

In this case, AI begins to do something other than help. It is able to write code, create pull requests and test cases. Teams begin to rely on AI to do real work.

Approximately 70% of tasks are to be supported by AI to transition to the next stage. There must be governance, the Knowledge Layer must be utilized correctly and teams must be at ease with AI.

C. Phase 3: AI-First (Months 10–18+).

AI systems do the majority of SDLC at this phase. Humans are involved but only to a greater extent as a control and decision-maker. This involves approval of architecture, code merge, staging sign-off, rollout control and handling of critical issues.

The developers are more of a reviewer and governor than a manual writer.

VIII. IMPLEMENTATION ROADMAP

To implement AI coding successfully, seven workstreams (3) must be coordinated to implement the code and follow the three adoption phases. Fig. 5 demonstrates the entire implementation roadmap including workstream activities and milestones.

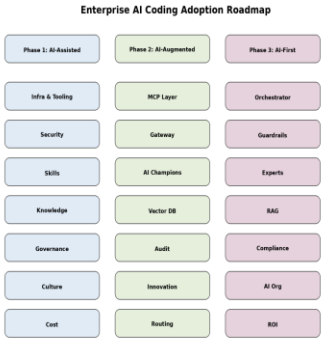

Figure 6: Enterprise AI Coding Adoption Implementation Roadmap with Seven Workstreams and Key Milestones

Infrastructure workstream builds off of IDE and source control configuration into MCP integration up to deployment of Central Orchestrator. The security progresses with the first review by AI tools to the deployment of DLP and up to the full six-layers security gateway. Skills are in the four level learning path. The Knowledge Base workstream will transition through

Knowledge Layer V1 and documentation audit to the self-learning Intelligent Layer. IP policy drafting is further developed to ongoing audit systems. Culture evolves out of executive sponsorship to the hackathons to governance-first organizational culture. The cost optimization is shifted to model benchmarking towards multi-model routing to ROI-optimized model mix.

The adoption journey has six major milestones: M1 (Pilot Launch), M2 (Phase 1 Graduation), M3 (Agent Rollout at scale), M4 (Phase 2 Graduation), M5 (Central Orchestrator Live), and M6 (Full AI-First Operational). The milestones are also cross-workstream synchronization points where the leadership evaluates the preparedness and then moves on.

IX. ENTERPRISE SCENARIO

GlobalBank is a fictitious financial services company that has 2,000 engineers in 150 teams and is subject to the Sarbanes-Oxley Act (SOX)(17), Payment Card Industry Data Security Standard (PCI-DSS), and General Data Protection Regulation (GDPR). An EACRF analysis indicates: Infrastructure 3.2, Skills 1.8, Processes 2.5, Governance 2.0, Culture 2.2 (composite 2.34) at the Phase 1 border with critical Skills and Governance gaps.

Phase 1 (4 months): security audit and DLP implementation, three pilot teams through Copilot and current review processes, AI IP policy development, Level 1-2 training implementation, documentation audit of readiness to use the Knowledge Base, and model cost benchmarking. Exit scores: Infrastructure 3.5, Skills 2.6, Processes 2.7, Governance 2.8, Culture 2.8 (composite 2.88).

Phase 2 (6 months): six-layer security gateway is active, AI agents are distributed among 20 teams, Knowledge Layer V1 with coding standards and API documentation, tiered review process, 50 AI Champions trained, multi-model routing deployed, audit trail tooling is live. Target scores: Infrastructure 4.0, Skills 3.5, Processes 3.5, Governance 3.8, Culture 3.5 (composite 3.66), positioning Phase 3 in 1218 months

X. DISCUSSION

The EACRF is an attempt to fill a potential practical gap in AI implementation in the software development lifecycle. Data security is one of the most crucial issues to organizations and the risk of data leakage is directly addressed by the six-layer model security testing framework (10). The framework prevents the departure of proprietary code and sensitive information without due validation by following a zero-trust approach. This simplifies the process of security teams giving the green light to use AI tools.

The next significant challenge is ensuring that AI tools are conscious of the organizational standards, policies and knowledge. The Intelligent Layer of centralized knowledge base alters the use of AI coding tools in an organization. Such tools do not exist as stateless assistants, but rather context-aware partners. The system is enhanced through semantic search, vector embeddings (2) and continuous learning based on the results of the past delivery and offers more value as the successive feature deliveries are made. Another typical cause of AI initiatives failure is the introduction of tools without properly training the workforce, which is addressed by the four-level employee learning path (6). The progression that is controlled by certification is the way to make sure that employees develop the needed skills before proceeding and the role-based learning approach enables companies to prioritize their training process in order to make sure that they concentrate on the most influential groups.

The multi-model cost routing approach demonstrates that the adoption of enterprise AI does not necessarily involve using costly models. The match of model capability with the complexity of the task enables the organization to save about 40-60 percent at the cost of preserving the desired quality. This renders the economic viability of implementing AI more reasonable and viable.

Although the EACRF is a viable way to go when an organization is interested in adopting AI in software development, there are possible limitations. The framework has been designed based on the challenges that most organizations encounter and the blistering development in AI. It is, however, yet to be tried at scale with a broad spectrum of enterprises. Due to this, the maturity levels and the relative significance of each dimension can be different in each organization. The framework also presupposes comparatively centralized enterprise environment, which does not necessarily represent highly distributed or unique organizations. This can be further developed as more organizations implement and test the framework on real-life situations.

XI. CONCLUSION

We have presented a framework in this paper, the name of which is Enterprise AI Coding Readiness Framework (EACRF) and which any organization can adhere to in case they have a vision of transforming their human-based legacy development to

AI-based software development. Most organizations are enthusiastic about the implementation of AI, but they have difficulty in real issues like security, skills, governance, and cost. The EACRF attempts to deal with these issues in an organized but pragmatic manner. It unites a six-layer zero-trust security model to secure sensitive data, a centralized body of knowledge containing an intelligent layer to make AI tools more context-sensitive, a four-level learning journey to train the workforce, and a multi-model routing strategy to control costs in an efficient way.

The implementation roadmap of seven works streams provides the organizations with a clear direction to follow. It indicates that such areas of the economy as infrastructure, security, skills, knowledge, governance, culture and cost cannot improve in isolation. By moving these regions to collide, organizations can be in a better position to prevent numerous pitfalls in early AI projects.

This is to be regarded as a step in the right direction to a larger vision of AI-based software development (1). In an earlier published paper, AI-First Software Development Lifecycle: An Agent-Driven Framework of Autonomous Planning, Coding, Testing and Deployment we explained what an ideal AI-first SDLC might be at the organizational level. The former paper concentrated on the destination whereas this paper concentrates on the journey- assisting organizations to know how they can slowly drift towards that vision in an orderly and controllable manner.

This is not meant to offer a flawless case that fits every one. Rather, it seeks to offer a point of departure that can be customized to suit the situation and preparedness of the organization. With additional businesses starting their AI adventure, the framework is bound to improve as it will be shaped by experience and lifelong learning.

A. ACKNOWLEDGEMENTS

The author thanks the anonymous reviewers for their constructive feedback on earlier versions of this manuscript.

- **Funding:** This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.
- **Contributions by authors:** Debashis Patra, Ambar Nath Saha conceived the framework, conducted the literature review, designed the readiness assessment model, model security testing framework, knowledge base architecture, employee learning path, cost analysis, adoption strategy, and implementation roadmap, developed the enterprise scenario, and wrote the manuscript.
- **Competing interests:** The author declares no competing interests.
- **Information and resource availability:** No datasets were generated or analysed during this study. All data supporting the findings are available within the article. No restrictions on materials apply.

X. REFERENCES AND NOTES

[1] Ambar Nath Saha, Debashis Patra, 2026. "AI-First Software Development Lifecycle: An Agent-Driven Framework for Autonomous Planning, Coding, Testing, and Deployment", *ESP Journal of Engineering & Technology Advancements* 6(1): 131-139.

[2] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. Pinto, J. Kaplan, H. Edwards, Y. Burda, and N. Joseph, Evaluating Large Language Models Trained on Code, arXiv preprint arXiv:2107.03374. 1(1) (2021) 1-34.

[3] N. Forsgren, J. Humble, and G. Kim, Accelerate: The Science of Lean Software and DevOps, IT Revolution Press, Portland, USA. 1(1) (2018) 1-288.

[4] E. Rogers, Diffusion of Innovations, 5th edition, Free Press, New York, USA. 5(1) (2003) 1-576.

[5] L. Tornatzky, M. Fleischer, and A. Chakrabarti, The Processes of Technological Innovation, Lexington Books, Lexington, USA. 1(1) (1990) 1-298.

[6] T. Fountaine, B. McCarthy, and T. Saleh, Building the AI-Powered Organization, Harvard Business Review. 97(4) (2019) 62-73.

[7] P. Vaithilingam, T. Zhang, and E. Glassman, Expectation vs. Experience: Evaluating the Usability of Code Generation Tools Powered by Large Language Models, ACM CHI Conference on Human Factors in Computing Systems. 1(1) (2022) 1-7.

[8] A. Ziegler, E. Kalliamvakou, X. Li, A. Rice, D. Rifkin, S. Simister, G. Sittampalam, and E. Aftandilian, Productivity Assessment of Neural Code Completion, Proceedings of the 6th ACM SIGPLAN International Symposium on Machine Programming. 1(1) (2022) 21-29.

[9] J. Kotter, Leading Change, Harvard Business Review Press, Boston, USA. 1(1) (2012) 1-208.

[10] G. Westerman, D. Bonnet, and A. McAfee, Leading Digital: Turning Technology into Business Transformation, Harvard Business Review Press, Boston, USA. 1(1) (2014) 1-292.

[11] J. Humble and D. Farley, Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation, Addison-Wesley Professional, Boston, USA. 1(1) (2010) 1-512.

[12] E. Brynjolfsson and A. McAfee, The Second Machine Age: Work, Progress, and Prosperity in a Time of Brilliant Technologies, W. W. Norton, New York, USA. 1(1) (2014) 1-336.

[13] A. Bacchelli and C. Bird, Expectations, Outcomes, and Challenges of Modern Code Review, IEEE International Conference on Software Engineering. 1(1) (2013) 712-721.
[14] Y. Jia and M. Harman, An Analysis and Survey of the Development of Mutation Testing, IEEE Transactions on Software Engineering. 37(5) (2011) 649-678.
[15] D. Anderson, Kanban: Successful Evolutionary Change for Your Technology Business, Blue Hole Press, Sequim, USA. 1(1) (2010) 1-278.
[16] European Parliament, EU Artificial Intelligence Act, Official Journal of the European Union. 1(1) (2024) 1-144.
[17] US Copyright Office, Copyright Registration Guidance: Works Containing Material Generated by Artificial Intelligence, Federal Register. 88(51) (2023) 16190-16194.
[18] Financial Industry Regulatory Authority (FINRA), Report on Artificial Intelligence in the Securities Industry, FINRA Report. 1(1) (2024) 1-30.
[19] G. Kim, J. Humble, P. Debois, and J. Willis, The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations, IT Revolution Press, Portland, USA. 2(1) (2021) 1-480.
[20] E. Schein, Organizational Culture and Leadership, 5th edition, Jossey-Bass, San Francisco, USA. 5(1) (2017) 1-416.