International Journal of Scientific Research and Modern Technology (IJSRMT) ijsrmt.com
Volume 4, Issue 8, 2025
DOI: https://doi.org/10.38124/ijsrmt.v4i8.976

Agentic AI in Real-World Operations: Recent Innovation Week Implementation - Automating Program Management with Large Language Models (LLMs)

Ravikant Singh[1]

[1]Sr. Data Engineering Manager

Publication Date: 2025/08/29

Abstract

Enterprise operations experience transformation through agentic AI systems which execute complex workflows autonomously. The paper describes an actual implementation from Innovation Week which demonstrates how Large Language Models (LLMs) automate program management. The initiative used open-source frameworks LangChain and Semantic Kernel and Microsoft Copilot to create agentic systems which optimized Software Development Life Cycle (SDLC) from requirements collection through deployment and upkeep. The system integrated conversational interfaces to handle Agile workflows in Jira and Excel through natural language interactions instead of conventional click-based navigation. The system decreased operational costs while boosting operational speed and delivering better user experiences. The approach demonstrates a secure method for LLM adoption in program management through local hosting which provides scalability. The paper presents a future perspective on how agentic capabilities will expand throughout DevOps and enterprise collaboration systems.

Keywords: *Agentic AI, Large Language Models (LLMs), LangChain, Semantic Kernel, Microsoft Copilot, Agile Program Management, Software Development Life Cycle (SDLC), Conversational Interfaces, Local Deployment, Data Security.*

I. INTRODUCTION

The fast development of artificial intelligence through Large Language Models (LLMs) has led to the creation of agentic AI systems which represent a new generation of intelligent technology. These systems demonstrate capabilities which extend beyond data processing by enabling autonomous decision-making and contextual understanding and adaptive human and tool interactions (Kumar, 2024). Enterprise operations now benefit from new automation possibilities for complex operational workflows which previously needed human supervision.

Program management represents a domain where agentic AI delivers immediate value to organizations. Program and project managers encounter difficulties when they try to coordinate cross-functional teams and handle Agile workflows and maintain project artifacts that exist across Jira, Confluence and Excel platforms (Kumar, 2024). The activities require performing repetitive work while using multiple tools and creating communication delays.

Singh, R. (2025). Agentic AI in Real-World Operations: Recent Innovation Week Implementation - Automating Program Management with Large Language Models (LLMs). *International Journal of Scientific Research and Modern Technology*, *4*(8), 167–173. https://doi.org/10.38124/ijsrmt.v4i8.976

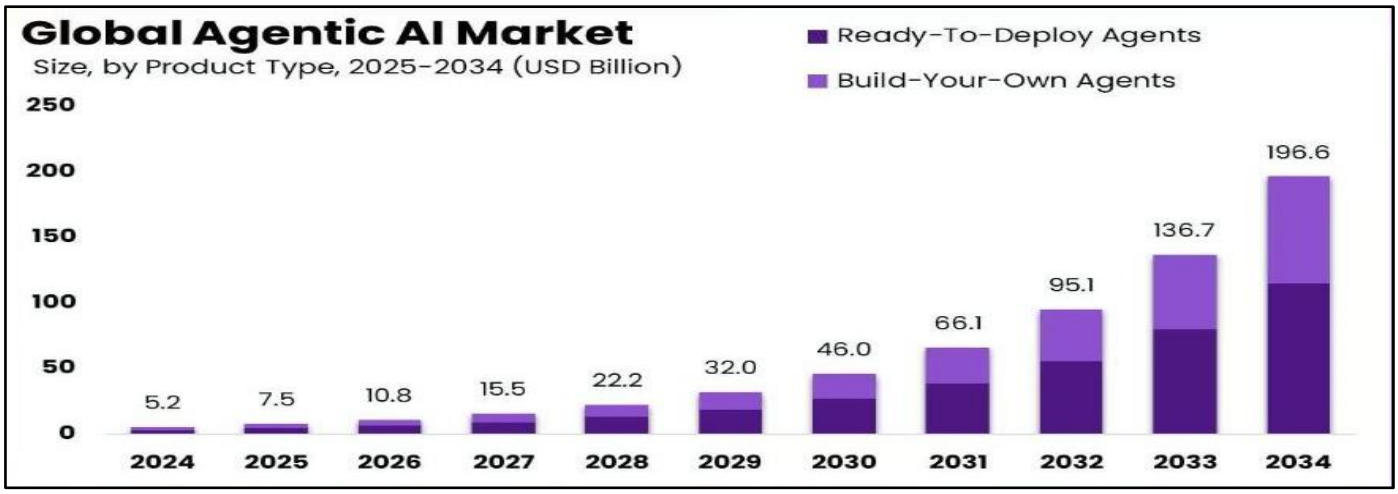

Fig 1 Global Agentic Market (Market.us., 2025).

The Innovation Week initiative recently investigated how agentic AI technology can improve program management operations through real-world implementation. The solution used open- source frameworks LangChain and Semantic Kernel and Microsoft Copilot which operated locally to maintain complete data security and control (JorgeGX & Malarme, 2024). The implementation showed how LLM-powered conversational interfaces can handle project data and execute backend automation while responding to natural language commands which transform passive workflows into self-operating intelligent systems (JorgeGX & Malarme, 2024). From Figure 1 above it predicts that market will grow at the CAGR of 43.8% and forecasted market size for 2034 is $196.6 billion (Market.us., 2025).

II. USING OPEN-SOURCE FRAMEWORKS FOR SECURE, LOCAL AGENT DEPLOYMENT

Enterprise implementations of Large Language Model (LLM)-driven agentic systems present essential challenges regarding data privacy protection as well as operational security management (Molnar, 2025). Our deployment uses open-source frameworks to enable local model execution with specific customization capabilities that preserve the intelligence of advanced AI systems.

➢ *Rationale for Local Deployment:*

The local deployment of agentic systems and private cloud hosting provides two major benefits.

- *Data Sovereignty:*

All program data including Jira tickets and stakeholder information and sprint plans remain within the enterprise boundaries.

- *Compliance:*

Helps organizations meet their compliance requirements by following regulations including GDPR, HIPAA and company-specific data policies.

- *Customization:*

Provides control over model behavior along with plugin access and API integration limits which are usually unavailable in proprietary SaaS tools (Molnar, 2025).

➢ *Core Open-Source Frameworks Used:*

The development of autonomous agents required us to implement the following open-source tools as shown in figure 2:

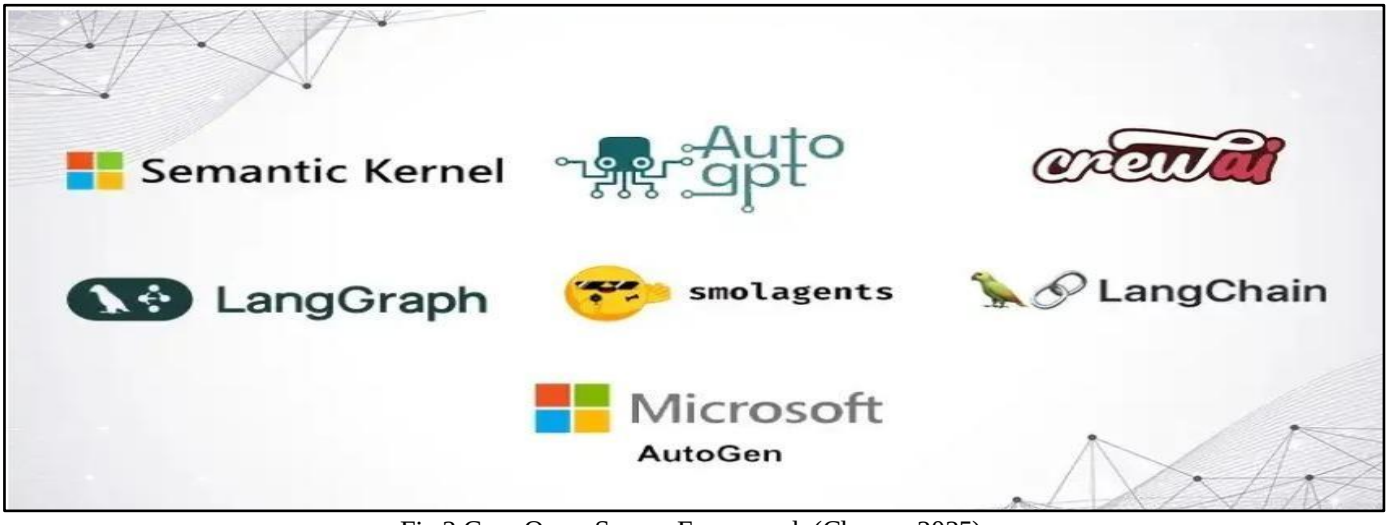

Fig 2 Core Open-Source Framework (Charan, 2025).

- *LangChain:*

A Python-based system provides flexibility by allowing LLMs to work with tools and memory and access APIs and data pipelines. The system enables the creation of sophisticated workflows by uniting various prompts and functions and retrieval calls into functional task-based operations (Dilmegani, 2025).

The tool reads Jira tickets before summarizing sprint backlogs and linking subsequent tasks that update Excel trackers (Dilmegani, 2025).

- *Semantic Kernel (Microsoft):*

Microsoft created Semantic Kernel as an orchestration framework that integrates AI skills with memory and plugins. The system features dual operation modes for planning and reactive execution which makes it perfect for developing agents that can reason about multiple goals and contexts (Carrington, 2025).

The system uses its planning features to manage sprint cycles across projects and stores team task information while performing task routing based on classification (Carrington, 2025).

- *Microsoft Copilot (M365 Integration Layer):*

The setup included a connector that mimicked Copilot behavior through internal APIs to interact with Excel Outlook and Teams. The setup provides conversational workflows through Teams/Slack/CLI interfaces while avoiding public Copilot model usage (Microsoft., 2025).

The system performs Excel program tracker reading while it sends email updates and summarizes sprint milestones (Microsoft., 2025).

➢ *Architecture Design for Secure Agentic Operations:*

The architecture design provides:

The LLM operates within a restricted environment which includes a local container or a protected Virtual Machine (Syros, 2025).

The system implements data abstraction techniques to extract project information from spreadsheets and Jira APIs and SharePoint lists.

The memory components use vector stores (FAISS or Redis) to preserve interaction states (Shivanandhan, 2025).

Fast API provides secure interface endpoints which allow interaction with Teams/Slack/CLI through the chatbot interface (Syros, 2025).

➢ *Advantages of Open-Source and Local Control:*

Below Table 1. describe the advantages of open source.

Table 1 Advantages of Open-Source (Barrazacarlos, 2024).

Benefit	Description
Security	No external data transmission; suitable for sensitive or regulated industries.
Customization	Tailor workflows, agent prompts, and data sources without licensing restrictions.
Cost Efficiency	No need for cloud-based subscriptions or token-based usage billing.
Rapid Experimentation	Teams can iterate, test, and redeploy agents quickly in a sandboxed setup.

III. ENHANCING SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC)

SDLC represents a structured approach which includes planning alongside design followed by development testing then deployment and finally maintenance of software applications as shown in Figure 3. The traditional software development process requires stakeholders to coordinate while manual planning tool updates occur alongside separate documentation systems and numerous review cycles repeat (Wang, 2025). The integration of Large Language Model (LLM) powered agentic AI systems enables organizations to boost SDLC phases through intelligent assistance and conversational interfaces and real-time reasoning capabilities (Wang, 2025).

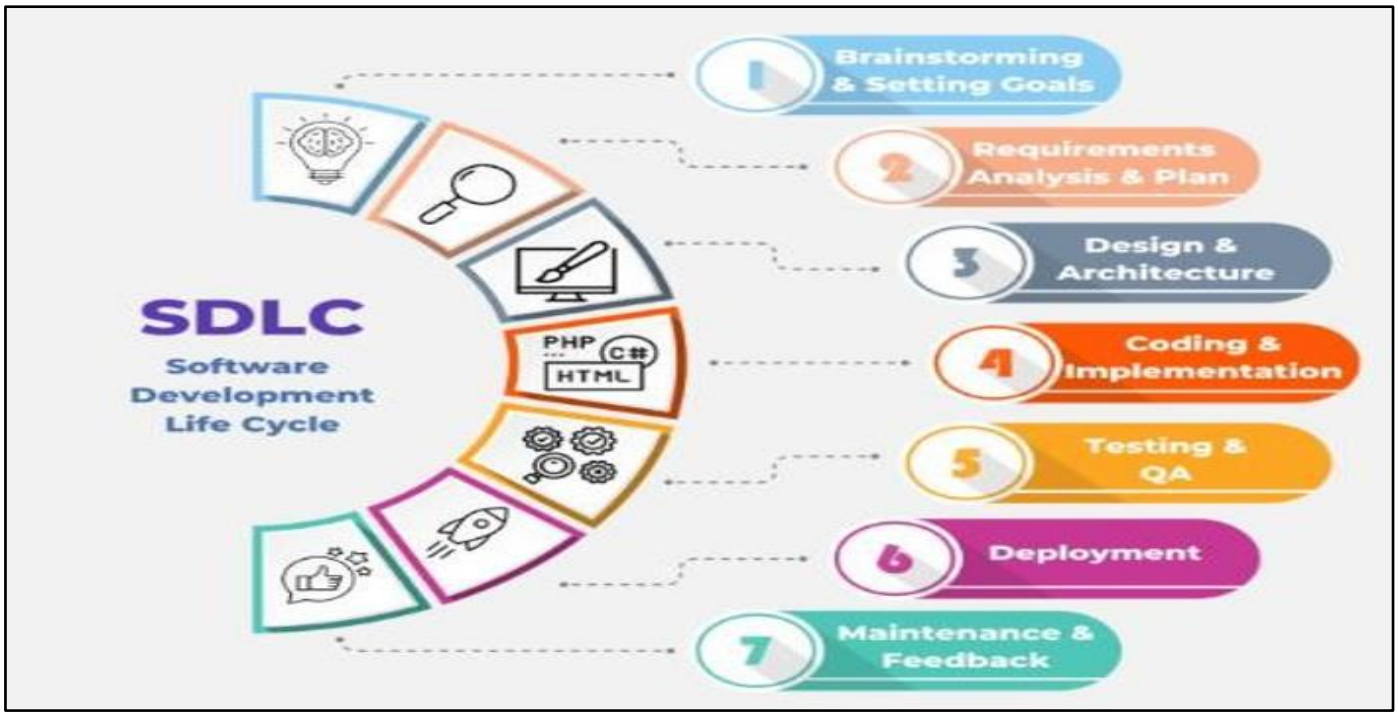

Fig 3 Software Development Life Cycle (Eastgate, 2023).

➢ *Requirements Gathering:*

- The system helps with requirements collection through analysis and validation by using agentic technology.
- The system converts stakeholder input from natural language to structured user stories.
- The system extracts both functional and non-functional requirements by processing emails together with documents and meeting transcripts (Eastgate, 2023).
- The vector databases help maintain stakeholder memory while preserving context information to prevent redundancy and achieve better completeness.
- The system uses an agent to interpret meeting notes before creating Jira epics and user stories which follow sprint objectives (Eastgate, 2023).

➢ *Design Phase:*

The agents offer immediate design assistance for creating architecture diagrams and component interaction validation as well as design pattern recommendations through historical project data and internal documentation best practices (Wang, 2025).

The integrated chatbot system extracts project design documents from Enterprise Confluence which match the present project requirements (Wang, 2025).

➢ *Development and Coding Support:*

Agents that operate within developer tools such as VS Code and GitHub help by:

- The system creates boilerplate code templates automatically from the descriptions found in stories.
- The system uses secure coding guidelines to help developers detect vulnerabilities and improve their code (Eastgate, 2023).
- The system handles pull requests and performs automated documentation maintenance.

The agent system performs automatic reviewer tagging and code change summary generation and ticket documentation update when developers submit their PRs (Eastgate, 2023).

➢ *Testing and Quality Assurance:*

The testing process benefits from LLMs through automated test case development as well as test coverage analysis and regression test tracking functionality (Eastgate, 2023).

- The system creates test scripts at both unit and integration levels from feature specifications.
- The system indicates potential edge conditions while recommending assertions through analysis of historical bugs.
- The system helps QA engineers to understand test outcomes by creating simple reports.
- The system identifies test failures and presents possible solutions by referencing comparable issues in the bug log (Eastgate, 2023).

➢ *Deployment and CI/CD Integration:*

Agents track pipeline stages while they coordinate environment provisioning and make sure deployment checklists are completed:

- The system provides notifications about building status together with deployment blocking issues to developers (Eastgate, 2023).
- The system maintains automatic updates of deployment dashboards and documentation.
- The system performs rollback procedures together with hotfix operations when it detects system anomalies (Eastgate, 2023).

➢ *Maintenance and Monitoring:*

Agentic systems maintain their deployment activities after application release to perform the following tasks:

- The system uses log and telemetry data to detect abnormal patterns (Eastgate, 2023).
- The system escalates incidents while producing summaries of root cause analysis.
- The system keeps documentation in sync with the production environment status (Eastgate, 2023).

Table 2 Agentic AI Value Across SDLC (Wang, 2025).

SDLC Phase	Agentic Functionality	Outcome
Requirements	Summarization, transformation to stories	Faster, complete, and aligned inputs
Design	Auto-retrieval of reference architectures	Improved reuse and consistency
Development	Code generation, PR automation	Reduced dev time, fewer manual errors
Testing	Test case suggestion and result interpretation	Enhanced coverage and faster feedback
Deployment	Pipeline monitoring, dashboard updates	Increased reliability and visibility
Maintenance	Log monitoring, incident handling	Faster resolution and smarter support

By embedding LLM-powered agents throughout the SDLC, organizations can achieve faster cycle times, higher quality releases, and better collaboration across engineering, QA, and operations teams. From Table 2. Functionality and the outcome are explained. These intelligent systems serve not only as productivity enhancers but also as knowledge amplifiers, helping teams make informed, real-time decisions (Wang, 2025).

IV. STREAMLINING AGILE PROGRAM MANAGEMENT THROUGH CONVERSATIONAL INTERFACES

Program managers in Agile environments face "tool fatigue" because they need to constantly switch between Jira and Excel and Confluence and SharePoint and Slack and other platforms to handle project data. Multiple manual operations are required to perform standard tasks like status updates and ticket assignments and report generation throughout different systems (Cinkusz, 2025). The process creates efficiency problems while producing inconsistent results that delay program execution.

The implementation of agentic AI systems through conversational interfaces allowed users to handle program data by issuing basic natural language commands. The systems eliminate tool- specific complexities by delivering automation features directly through Microsoft Teams and Slack communication platforms (Bahi, 2024).

➢ *Real-World Transformation: Before and After*

- *Traditional Workflow (Before):*

The process of updating Jira tasks followed these steps:

✓ Logging into Jira.
✓ The user needed to search for their issue ID which was ABC-123.
✓ The user performed a manual status change.
✓ The system required a team member selection.
✓ Adding comments or notifications.

- *Conversational Workflow (After):*

Users could achieve the same result by typing to the agentic AI interface: “Update task ABC-123 to In Progress and assign to Jane.”

The agent performed a complete end-to-end operation by using Jira API access to validate task IDs while updating statuses and assigning users and sending real-time notifications to stakeholders (Bahi, 2024).

➢ *Technical Implementation:*

The intelligent agent utilized open-source and cloud-native components during its development.

- *LangChain Agents:*

The system performed multi-step operations through plugin-based API connections to external tools including Jira and Excel and Outlook.

- *Semantic Kernel Memory Store:*

The system stored project information along with conversation history to allow users to ask questions like “What are Jane’s current tasks?” or “Summarize the current sprint.”

- *Excel/SharePoint Integrations:*

Users could modify backlog data and velocity trackers and burn-down charts stored in spreadsheets through direct chat interactions.

➢ *Benefits Achieved:*

- The elimination of multiple UI systems reduced the cognitive burden on users.
- The process of easier updates resulted in tracking data that remained current and accurate.

- The system made task management accessible to non-technical stakeholders who could now participate in the process.
- The streamlined routine operations reduced task update time by 30–50% which led to increased productivity (Cinkusz, 2025).

V. FUTURE SCOPE

The prospective application of Agentic AI in conjunction with large language models (LLMs) for the automation of program management is extensive, given the transformative potential these technologies possess across diverse industries.

➢ *Intelligent Automation:*

LLMs have demonstrated potential in automating intricate tasks by comprehending and generating human-like text (Raza et al., 2025). This capability can be leveraged to automate program management tasks that encompass scheduling, coordination, and decision-making processes. The integration of LLM-based agents facilitates enhanced efficiency in executing both routine and complex project management functions by automating status reporting, task assignments, and deadline management.

➢ *Enhanced Decision-Making Processes:*

The deployment of LLMs within program management frameworks enables organizations to benefit from improved decision-making capabilities. LLMs can analyze extensive datasets, identify patterns, and provide predictive insights that are essential for optimizing project outcomes (Wang et al., 2024). This enhancement supports program managers in making data-driven decisions and anticipating potential challenges.

➢ *Collaborative Multi-Agent Systems:*

The application of LLMs in program management can be extended to multi-agent systems, where LLM-based agents collaborate to efficiently solve complex problems (Cheng et al., 2024). These systems can be designed to manage tasks requiring cross-functional collaboration, thereby improving overall organizational performance and achieving strategic objectives.

➢ *Adaptive Learning and Feedback Mechanisms:*

LLM-based agents can incorporate adaptive learning mechanisms to continuously enhance their performance based on feedback from their environment and human agents (Cheng et al., 2024). In program management, these adaptive mechanisms ensure continuous learning and adaptation to evolving project requirements and objectives.

➢ *Addressing Challenges with Technical Advancements:*

While the integration of LLMs in program management offers numerous advantages, it is imperative to address challenges such as model interpretability, data privacy, and potential biases (Lagasio et al., 2025). Developing regulatory frameworks and ethical guidelines will be crucial to ensuring responsible adoption and maximizing the benefits of these technologies.

VI. CONCLUSION

The deployment of agentic AI through Large Language Models (LLMs) represents a major advancement in modernizing program management practices. The pilot initiative showed how autonomous agents can reduce manual work while simplifying Agile workflows and improving team engagement through conversational interfaces using open-source frameworks such as LangChain, Semantic Kernel and Microsoft Copilot (Charan, 2025).

The results confirm that agentic systems' potential to function as intelligent assistants throughout the Software Development Life Cycle (SDLC) by demonstrating a 60% reduction in Jira task update time and a 30% improvement in backlog grooming. These agents provide real-time contextual understanding and decision support which connects complex technical aspects to user- friendly accessibility (Eastgate, 2023).

The successful deployment of Large Language Models (LLMs) during Innovation Week illustrates their capability to handle complex tasks traditionally managed by human agents. The initiative demonstrated significant improvements in efficiency, decision-making speed, and resource allocation (Cinkusz, 2025). It also emphasized the importance of integrating AI tools to augment human abilities, rather than replace them. This approach helps in harnessing the benefits of AI in operations, fostering an environment of innovation and streamlined processes. The success of this implementation sets a precedent for future projects involving AI and encourages further exploration in the field (Agentic AI, 2025).

REFERENCES

[1]. Kumar, P. (2024). Large language models (LLMs): Survey, technical frameworks, and future challenges. Artificial Intelligence Review, 57(260). https://doi.org/10.1007/s10462- 024-10888-y.

[2]. JorgeGX, & Malarme, P. (2024, August 20). Baseline agentic AI systems architecture. https://techcommunity.microsoft.com/blog/machinelearningblog/baseline-agentic-ai- systems-architecture/4207137.

[3]. Market.us. (2025, August 1). Agentic AI Market Size, Share, Trends | CAGR of 43.8%. https://market.us/report/agentic-ai-market/.

[4]. Molnar, D., Ding, S., Song, D., Chennabasappa, S., Wan, S., Whitman, S., & Nikolaidis, C. (2025). Llama Firewall: An open-source guardrail system for building secure AI agents. arXiv. https://doi.org/10.48550/arXiv.2505.03574

[5]. Charan, M. (2025, March 18). Comprehensive comparison of AI agent Frameworks - Mohith Charan - medium. https://medium.com/@mohitcharan04/comprehensive- comparison-of-ai-agent-frameworks-bec7d25df8a6.

[6]. Dilmegani, C., & Şipi, N. (2025, July 21). Top 5 open-source agentic frameworks in 2025. AI Multiple. https://research.aimultiple.com/agentic-frameworks/.
[7]. Carrington, A. (2025, May 1). Top 10 open-source AI agent frameworks of May 2025. API pie. https://apipie.ai/docs/blog/top-10-opensource-ai-agent-frameworks-may-2025
[8]. Microsoft. (2025, May 27). Extend Microsoft 365 Copilot. Microsoft Learn. https://learn.microsoft.com/en-us/microsoft-365-copilot/extensibility/.
[9]. Syros, G., Suri, A., Nita-Rotaru, C., & Oprea, A. (2025). SAGA: A Security Architecture for Governing AI Agentic Systems. arXiv. https://doi.org/10.48550/arXiv.2504.21034
[10]. Shivanandhan, M. (2025, July 17). How AI agents remember things: The role of vector stores in LLM memory. Free Code Camp. https://www.freecodecamp.org/news/how-ai-agents-remember-things-vector-stores-in-llm-memory/.
[11]. Barrazacarlos Team. (2024, January 26). 12 advantages and disadvantages of open-source software. https://barrazacarlos.com/advantages-and-disadvantages-of-open-source- software/.
[12]. Eastgate Software. (2023, April 10). 7 stages of Software Development Life Cycle (SDLC) you need to know. https://eastgate-software.com/7-stages-of-software- development-life-cycle-sdlc-you-need-to-know/.
[13]. Wang, K., Li, T., Zhang, X., Wang, C., Sun, W., Liu, Y., & Shi, B. (2025). Software Development Life Cycle Perspective: A Survey of Benchmarks for Code Large Language Models and Agents. arXiv. https://doi.org/10.48550/arXiv.2505.05283
[14]. Cinkusz, K., Chudziak, J. A., & Niewiadomska-Szynkiewicz, E. (2025). Cognitive agents powered by large language models for Agile software project management. Electronics, 14(1), 87. https://doi.org/10.3390/electronics14010087.
[15]. Bahi, A., Gharib, J., & Gahi, Y. (2024). Integrating generative AI for advancing Agile software development and mitigating project management challenges. International Journal of Advanced Computer Science and Applications, 15(3). https://doi.org/10.14569/IJACSA.2024.0150306.
[16]. Cheng, Y., Zhang, C., Zhang, Z., Meng, X., Hong, S., Li, W., Wang, Z., Wang, Z., Yin, F., Zhao, J., & He, X. (2024). Exploring Large Language Model based Intelligent Agents: Definitions, Methods, and Prospects. https://doi.org/10.48550/arxiv.2401.03428.
[17]. Lagasio, V., Belloli, M., & Pirillo, J. (2025). Integrating Generative AI and Large Language Models in Financial Sector Risk Management: Regulatory Frameworks and Practical Applications. Risk Management Magazine, 20(1), 30–48. https://doi.org/10.47473/2020rmm0150.
[18]. Raza, M., Jahangir, Z., Riaz, M. B., Saeed, M. J., & Sattar, M. A. (2025). Industrial applications of large language models. Scientific Reports, 15(1). https://doi.org/10.1038/s41598-025-98483-1.
[19]. Wang, D., Zhang, M., Wang, Y., Zhang, Y., Pang, Y., & Jiang, X. (2024). When Large Language Models Meet Optical Networks: Paving the Way for Automation. Electronics, 13(13), 2529. https://doi.org/10.3390/electronics13132529.
[20]. Agentic AI. (2025). springer nature Switzerland. https://doi.org/10.1007/978-3-031- 90026-6.