AI-Driven Development Life Cycle (AI-DLC) Method Definition

Rick-AP, Amazon Web Services

I. CONTEXT

The evolution of software engineering has been a continuous quest to enable developers to focus on solving complex problems by abstracting away lower-level, undifferentiated tasks. From early machine code to high-level programming languages and the emergence of frameworks, each advancement has boosted developer productivity. Now, the integration of Large Language Models has revolutionized how software is created, introducing conversational coding, automating interactions for tasks like code generation, bug detection, and test generation. This marks the **AI-Assisted** era, where AI enhances such fine-grained tasks.

As AI evolves, its applications are expanding beyond code generation to include requirements elaboration, planning, task decomposition, design and real-time collaboration with developers. This shift is kick-starting the **AI-Driven** era, where AI actively orchestrates the development process. But, existing software development methods, designed for human-driven, collocated teams, struggle to accommodate AI's speed, scale, flexibility, and advanced capabilities (ex. agents). Their reliance on manual workflows and rigid role definitions limits the ability to fully leverage AI. Bolting-on AI into these methods not only limits its potential, but also reinforces outdated inefficiencies. To fully leverage AI's transformative power, SDLC methods need to be reimagined. This reimagination requires AI to be a central collaborator, assigning tasks, roles, and iterations to enable faster decision-making, seamless task execution, and continuous adaptability.

This paper introduces and defines the **AI-Driven Development Lifecycle (AI-DLC)**, a reimagined, AI-native methodology that aims to fully integrate the capabilities of AI, setting the foundations for the next evolution in software engineering.

II. KEY PRINCIPLES

The principles in this section form the foundation for defining AI-DLC, shaping its phases, roles, artifacts, and rituals. These assumptions are critical for validating the proposed method, as they provide the underlying rationale behind its design.

1. REIMAGINE RATHER THAN RETROFIT

We choose to reimagine a development method rather than keeping the existing methods like SDLC or Agile (e.g., Scrum) and **retrofitting** AI into them. These traditional methods were built for longer iteration durations (months and weeks), which led to rituals like daily standups and retrospectives. In contrast, proper application of AI leads to rapid cycles, measured in hours or days. This needs continuous, real-time validation and feedback incorporation to ensure AI's output stays aligned with the intent and relevant. Should effort elimination, as the only goal, be as critical if AI dominates the boundaries between simple, medium and complex? Would teams like velocity be as relevant or should we start replacing it with Business Value, as an example? Also, AI is exceptionally evolving to automate manual processes including planning, task decomposition, requirements analysis, application design, coding, testing, and deployment. This automation raises the question of phases & iterations, their purpose and duration. These new dynamics warrant a **reimagination based on first principles thinking**, rather than a retrofitting. We need automobiles and not the faster horse chariots.

2. REVERSE THE CONVERSATION DYNAMICS

AI-DLC introduces a fundamental shift where **AI initiates & directs the conversations** with humans instead of humans initiating and directing AI as in the current paradigm. This reversal enables the leading AI to systematically extract details by implementing a new business function) into actionable tasks, generating recommendations, and proactively identifying AI's human serve as approvers, validating, selecting options, and confirming outcomes. This shift allows AI to leverage its extensive knowledge to focus on high-value human interactions while AI handles planning, task decomposition, and automation. By reversing the traditional dynamic, AI-DLC ensures that human involvement is purposeful, concentrating on oversight, risk mitigation, and strategic alignment, thereby enhancing both velocity and quality. An analogy to illustrate this is Google Maps-guided navigation versus asking for random directions. Conversation initiating directions (AI's task decomposition and recommendations) Along the way, humans maintain oversight and moderate the journey as needed.

3. INTEGRATION OF DESIGN TECHNIQUES INTO THE CORE

Agile frameworks like Scrum or Kanban leaves the design techniques (ex. Domain Driven Design) out of scope and recommends the teams to choose their own. This has left critical decisions to be made by the teams without sufficient guidance. Design quality issues in US alone was estimated to cost $2.41 Trillion in 2022 (study). Rather than decoupling design techniques out, AI-DLC will have them as its integral part. There will be different flavors of AI-DLC for teams following Domain Driven Design (DDD), Behavior Driven Development (BDD) or Test Driven Development (TDD) respectively. This paper discusses the DDD

flavor of AI-DLC which will use DDD principles to break down systems into independent, right-sized bounded contexts that can be developed by AI-DLC units. This approach leverages DDD techniques during planning and task decomposition, ensuring developers only to validate and adjust. This integration is key to enabling hours-to-day iteration cycles while eliminating manual heavy-lifting and maintaining software quality (the mantra of "Build Better Systems Faster").

4. ALIGN WITH AI CAPABILITY

This paper is opinionated about the future potential of AI but fully acknowledges that AI is not perfect or at its full potential. AI-DLC is advancing but not yet reliable in autonomously handling high-level intentions into executable code or independently operating without human oversight, while also **maintaining interpretability and safety**. At the same time, the **AI-Assisted** paradigm of AI-DLC ensures that even if AI's capabilities plateau, the framework still remains providing significant value. It aims to fully unlock the full potential of AI as in development. AI-DLC adapts the **AI-Driven** paradigm, which balances AI's involvement with the capabilities and limitations of current AI. In AI-DLC, AI manages the overall workflow, recommends next steps, **decision-making, and execution**. This balance ensures that the strengths of AI are leveraged effectively without compromiting the critical safeguards provided by human oversight.

5. CATER TO BUILDING COMPLEX SYSTEMS

AI-DLC caters to building systems that demand continuous functional adaptability, **high architectural complexity, numerous trade-offs management**, scalability, integration and customization requirements. These necessitate the application of advanced design techniques, patterns, and best practices, ensuring the resulting AI-generated code base maintains long-term maintainability and manageability. Simpler systems that are developed by non-developer personas that needed few or no trade-off management are outside the scope of AI-DLC and are better suited for low-code/no-code approaches.

6. RETAIN WHAT STANDOUT HUMAN SYNERGIES

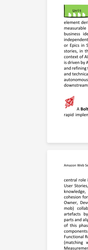
While reimagining the method, we will retain the artifacts and touchpoints from the existing methods that are critical for human collaboration. This includes "Business Intent" as a primary input, humans' and AI's understanding of what needs to be built, acting as well-defined contracts. We will retain user stories as they are in the re-imagined method also. Another example is the Risk Register that ensures AI-generated plans and codes comply with organizational standards. These retained elements will be then optimized for real-time use, allowing rapid iterations without compromising alignment or safety.

7. FACILITATE TRANSITION THROUGH FAMILIARITY

AI capabilities while maintaining human control over critical decisions.

III. CORE FRAMEWORK

This section outlines the core framework of AI-DLC, detailing its phases, roles, workflows, and key artifacts.


AI-DLC
Inception
Construction
Operations
Business Intent
Inception Phase
Construction Phase
Operations Phase
Business Intent
Business Tasks
Domain Models
Logical Design
Code
Deployment Artifacts
Production Systems
Feedback
Feedback
Feedback