MAC0499 — Trabalho de Formatura Supervisionado

Projeto: CD4AI — Regression Testing for AI-Enabled Systems

Orientador: Renato Cordeiro Ferreira
Coorientador: Prof. Alfredo Goldman vel Lejbman
Aluno: João Paulo Pereira da Silva
Número USP: 13695000

1 Introdução

A adoção massiva de *Large Language Models* (LLMs) viabilizou uma nova classe de sistemas de software — os *AI-Enabled Systems* — nos quais parte do comportamento é delegada a componentes de inteligência artificial. Esses sistemas evoluem continuamente: prompts de agentes são ajustados, *guardrails* são adicionados, workflows são reorganizados, tudo no mesmo ritmo de entregas de software convencional.

Essa velocidade de mudança de sistemas probabilísticos, ainda mais acentuada em ambientes de desenvolvimento ágeis, impõe problemas inéditos ao tentar conciliar robustez, agilidade e testabilidade em pipelines de *Continuous Delivery*. Os fluxos tradicionais de *Machine Learning* dependem de grandes conjuntos de dados pré-curados e ciclos longos de avaliação [5], então não acompanham o ritmo de iteração do desenvolvimento ágil. Ao mesmo tempo, suítes de testes sintéticos com frequência não capturarão situações imprevistas do uso real. E mesmo recorrendo ao comportamento observado em produção, percorrer todas as execuções não é viável em sistemas que operam em escala.

1.1 Objetivo

O objetivo deste trabalho é responder à seguinte questão:

> Como evoluir *AI-Enabled Systems*, em escala, de forma ágil, sem abrir mão de confiabilidade e robustez?

Acreditamos que podemos encontrar uma resposta por meio do padrão CD4AI, que estabelece um ciclo de vida em três estágios:

1. **Teste:** executa *passo do pipeline* de *Continuous Delivery* que condiciona a implantação à aprovação na suíte de regressão.
2. **Monitoramento:** executa coleta dos dados provenientes das execuções de *workflows* ou *agentes* em produção.
3. **Curadoria:** executa filtragem das execuções que, durante o monitoramento, mostraram problemas ou regressões. Esses dados são usados pela equipe de desenvolvimento para a criação de correções e novos testes.

A dinâmica entre esses estágios está ilustrada na Figura 1.

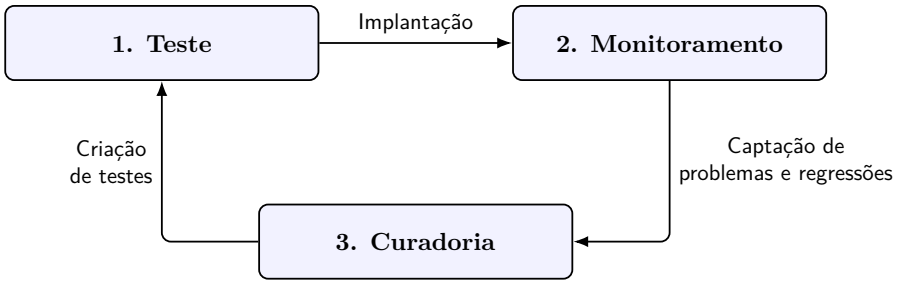

Figura 1: Dinâmica do padrão CD4AI.

1.2 Estrutura

Este documento está organizado da seguinte forma. A Seção 2 apresenta os conceitos fundamentais necessários para compreender o trabalho: CI/CD, *AI-Enabled Systems*, agentes e workflows, e teste de regressão. A Seção 3 descreve a metodologia adotada para responder à questão central. A Seção 4 descreve o plano de trabalho e o cronograma previsto.

2 Fundamentos

Apresentamos nesta seção os conceitos básicos necessários para compreender os detalhes sobre o padrão proposto.

2.1 Continuous Integration e Continuous Delivery

Continuous Integration (CI) é a prática de integrar frequentemente as alterações dos desenvolvedores em uma *branch* principal compartilhada, com cada integração validada por um *pipeline* automatizado de build e testes. *Continuous Delivery* (CD) estende a prática de CI ao criar formas de garantir que toda alteração, aprovada por um *pipeline de integração*, esteja sempre de fato apta a ser implantada em produção. Juntas, as práticas de CI/CD sustentam os ciclos de entrega de alta velocidade que caracterizam a engenharia de software moderna [2].

2.2 AI-Enabled Systems, Workflows e Agentes

AI-Based Systems são sistemas de software que incorporam componentes de Inteligência Artificial. Esses sistemas aprendem analisando o ambiente em que operam e tomam ações com o objetivo de exibir comportamento inteligente [4]. Neste trabalho, vamos nos referir a esses sistemas como *AI-Enabled Systems*.

Dentro desses sistemas, definimos dois tipos de fluxos: *workflows*, onde LLMs são orquestradas por trechos de código determinísticos, e *agentes*, em que, com mais liberdade, a própria LLM define dinamicamente, usualmente com loops de feedback, os processos a serem seguidos na resolução de uma tarefa [6].

2.3 Teste de Regressão

Teste de software é a atividade de avaliar um sistema em busca de defeitos. O teste de regressão é a prática de testar funcionalidades já implementadas, garantindo que mudanças no sistema não violem comportamentos existentes [3, 1]. Comumente vinculado a pipelines de *Continuous Delivery*, a suíte de regressão cresce à medida que novos defeitos são descobertos, solidificando cada correção como

uma restrição permanente. No caso de *AI-Enabled Systems*, testes de regressão podem garantir que defeitos percebidos em produção (e.g., um *agente* se comportando de maneira errônea) não voltem a ocorrer após melhorias de prompt.

3 Metodologia

Para responder à questão central deste trabalho, a metodologia é organizada em três etapas:

3.1 Estudo de Casos

Levantamento de casos de uso de *AI-Enabled Systems* no mercado e na literatura de padrões semelhantes, com foco em como equipes conciliam agilidade e confiabilidade nesses sistemas. Como parte deste estudo, o padrão CD4AI será aplicado em um caso real, permitindo observar suas limitações e oportunidades de melhoria em condições práticas.

3.2 Refinamento do Padrão

Com base nos casos estudados, o padrão será revisado e ajustado. O foco é identificar o que o CD4AI atual não cobre, o que precisa ser reformulado e quais novos elementos devem ser incorporados.

3.3 Definição Formal do Padrão

Consolidação do padrão refinado em sua forma definitiva, documentado segundo a estrutura canônica de padrões de software: intenção, motivação, problema, forças, solução e dinâmica.

4 Plano de Trabalho

		Meses (2026)								
	Etapas	04	05	06	07	08	09	10	11	12
1	Revisão bibliog. e levantamento	**X**	**X**	**X**	**X**					
2	Estudo de implementações existentes		**X**	**X**	**X**	**X**				
3	Refinamento do padrão					**X**	**X**			
4	Definição formal do padrão						**X**	**X**		
5	Escrita da monografia	**X**	**X**	**X**	**X**	**X**	**X**	**X**	**X**	
6	Apresentação									**X**

Figura 2: Cronograma das atividades.

Adicionalmente, no momento, os autores deste projeto estão tentando submetê-lo para a bolsa de empreendedorismo da USP. Caso seja aceito, parte do projeto será desenvolvido na Jheronimus Academy of Data Science (JADS), parte da Eindhoven University of Technology (TUe), na Holanda.

5 Bibliografia

Referências

[1] Martin Fowler. *Refactoring: Improving the Design of Existing Code*. Addison-Wesley Professional, 2nd edition, 2018.

[2] Martin Fowler. Continuous integration. https://martinfowler.com/articles/continuousIntegration.html, 2024.

[3] Jez Humble and David Farley. *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*. Addison-Wesley Professional, 1st edition, 2010.

[4] Silverio Martínez-Fernández, Justus Bogner, Xavier Franch, Marc Oriol, Julien Siebert, Adam Trendowicz, Anna Maria Vollmer, and Stefan Wagner. Software engineering for AI-based systems: A survey. *ACM Transactions on Software Engineering and Methodology*, 31(2), 2022.

[5] Danilo Sato, Arif Wider, and Christoph Windheuser. Continuous delivery for machine learning. https://martinfowler.com/articles/cd4ml.html, 2019.

[6] Erik Schluntz and Barry Zhang. Building effective agents. https://www.anthropic.com/research/building-effective-agents, 12 2024.