Encenando Modelos de Processo de Software em Sala: um Relato de Experiência com Alunos de Graduação

Talita Vieira Ribeiro[1,2], Thiago Silva de Souza[2] e Rigel Procópio Fernandes[2]

[1]Universidade do Estado do Rio de Janeiro (UERJ) – Rio de Janeiro, RJ – Brazil

[2]Instituto Brasileiro de Mercado de Capitais (IBMEC)

talita.ribeiro@ime.uerj.br, {talita.ribeiro, thiago.souza, rigel.fernandes}@professores.ibmec.edu.br

Abstract. *Context: Software Engineering (SE) students struggle with abstracting software process models due to a lack of practical experience. Objective: To report an Active Learning dynamic designed to help differentiate these models. Method: An enactment of the process models using role-playing and an image assembly kit was applied in an introductory SE course with 5 students. The dynamic used three rounds, imposing the constraints of each process model under analysis. Results: The Sequential model highlighted team idleness, late rework, and bottlenecks. The Incremental model demonstrated fluidity, resource optimization, and early value delivery. The Evolutionary model illustrated architectural risk and the management of uncertainty in initial decisions.*

Resumo. *Contexto: Alunos de Engenharia de Software (ES) enfrentam dificuldade na abstração dos modelos de processo de software por carência de vivência prática. Objetivo: Relatar uma dinâmica de Aprendizagem Ativa planejada para auxiliar na diferenciação desses modelos. Método: Uma encenação dos modelos de processo usando role-playing e kit de montagem de imagens foi aplicada em uma disciplina inicial de ES com 5 alunos. A dinâmica utilizou três rodadas impondo as restrições de cada modelo de processo sob análise. Resultados: O modelo Sequencial evidenciou ociosidade da equipe, retrabalho tardio e gargalos. O Incremental demonstrou fluidez, otimização de recursos, entrega de valor antecipado. O Evolutivo ilustrou o risco arquitetural e a gestão da incerteza nas decisões iniciais.*

1. Introdução

A Engenharia de Software (ES) é uma disciplina fundamental no currículo da ciência da computação, mas frequentemente apresenta um desafio pedagógico significativo: a dificuldade dos alunos de graduação em compreenderem seus conceitos abstratos e processos complexos sem terem tido experiência prática prévia, seja por meio de estágios ou projetos reais [1] [2]. A própria organização de um processo de software em fases (e.g. comunicação, planejamento, modelagem, construção e entrega [3]) já impõe uma carga

Cadernos do IME - Série Informática
e-ISSN: 2317-2193 (online)
DOI: 10.12957/cadinf.2026.95289

de abstração e de encadeamento lógico que costuma ser mais bem assimilada quando confrontada com situações concretas de projeto.

Um tópico particularmente propenso à confusão é a distinção entre os principais tipos de modelos de processos — sequencial, incremental e evolutivo —, uma vez que depende justamente de abstrair como as fases se relacionam e como (ou quando) o produto parcial é entregue. Em termos curriculares, tratamos esses modelos como representações do ciclo de vida: sequenciais (e.g. cascata e V) com fluxo linear e forte documentação; incrementais (e.g. iterativo incremental e RAD), com entregas por estágios operacionais; e evolutivos (e.g. evolucionário), com versões sucessivas e *feedback* rápido. A diferenciação conceitual entre "entregas operacionais por estágio" (incremental) e "versões que evoluem o entendimento e a solução" (evolutivo) aparece recorrentemente como ponto crítico para alunos iniciantes. A sua representação puramente teórica nos *slides* de aula frequentemente falha em consolidar um entendimento prático das consequências de cada escolha de modelo [4].

Além disso, observa-se uma dificuldade em manter a atenção em aulas expositivas tradicionais. A literatura recente em Educação em Computação aponta preferência de estudantes da nova geração (*Millennials* e *Post-Millennials*) por aprendizagem ativa, com trabalho cooperativo e atividades experienciais, que tendem a elevar engajamento e resultados quando comparadas a exposições passivas [5] [6]. Essa mudança de preferência pedagógica tem sido discutida em fóruns da área e motiva reconfigurações de práticas docentes no contexto de ES. Abordagens de aprendizado ativo, como dinâmicas baseadas em jogos ou peças (como LEGO), têm se mostrado eficientes para demonstrar conceitos complexos e abstratos de forma mais simples e tangível, aumentando o envolvimento dos alunos [7] [8] [9].

Pensando nessas problemáticas e características, adotamos a metodologia ativa para posicionar os estudantes no centro da aprendizagem. Isso foi realizado por meio de uma dinâmica de encenação de modelos de processo, na qual grupos simulam as restrições e fluxos de três variações — sequencial, incremental e evolutiva — utilizando papéis, requisitos, cartas e kits de montagem.

Este artigo tem como objetivo relatar a experiência de aplicar essa dinâmica em uma disciplina inicial de ES. O relato visa documentar a concepção detalhada da dinâmica, sua execução e os achados práticos observados. Pretende-se, assim, prover um roteiro replicável da dinâmica que auxilia na compreensão da estrutura e das relações das fases do ciclo de vida de software em diferentes modelos de processo. O artigo está organizado como segue: a Seção 2 apresenta o contexto e a metodologia; a Seção 3 detalha a implementação prática e os achados da execução; a Seção 4 traz a análise reflexiva sobre a dinâmica, focando em problemas e dificuldades encontradas; a Seção 5 aborda as limitações metodológicas e algumas considerações sobre possíveis replicações; e a Seção 6 apresenta as considerações finais.

2. Contexto e Metodologia

Esta seção detalha o contexto acadêmico, o raciocínio pedagógico e a metodologia utilizada para a concepção, execução e análise da dinâmica sobre modelos de processo de software.

2.1 Contexto Institucional e da Turma

2.1.1 Caracterização da Disciplina e da Turma

A dinâmica relatada foi aplicada na disciplina de Padrões e Projeto de Software, oferecida em uma instituição de ensino privada – Centro Universitário Ibmec. Esta disciplina, com carga horária de 60 horas, é cursada pelos alunos no terceiro semestre do Curso de Ciência de Dados e Inteligência Artificial, sendo equivalente a uma disciplina inicial de ES. O conteúdo programático da disciplina abrange processos de software, engenharia de requisitos e modelagem UML. A disciplina está inserida em um semestre temático focado em *back-end*, com outras disciplinas correlatas como Banco de Dados, Programação Orientada a Objetos, Modelagem Computacional e um Projeto Extensionista de construção de *back-end* de um software real.

O formato da oferta é presencial, e as aulas ocorrem em um laboratório equipado com projetor e computadores disponíveis para todos os alunos. O tempo total de cada encontro de aula é de 4 tempos de 50 minutos cada. Quanto aos participantes, a turma é pequena, composta por um total de 6 alunos, dos quais 5 estavam presentes no dia da aplicação da dinâmica. O perfil dos estudantes é de iniciantes, visto que nenhum aluno está estagiando ou possui experiência prática com projetos de software reais fora do ambiente universitário.

2.1.2 Conteúdo e Objetivos de Aprendizagem

O conteúdo principal abordado pela dinâmica proposta é a distinção e aplicação prática dos modelos de processos de software: sequencial, incremental e evolutivo. Este tópico é estrategicamente posicionado como tema inicial da disciplina de Padrões e Projeto de Software, servindo como base para a posterior discussão sobre a engenharia de requisitos e modelagem UML.

A atividade foi desenhada para atingir os seguintes objetivos de aprendizagem de nível prático e analítico:

- **Diferenciar o fluxo de trabalho:** Ser capaz de distinguir, na prática, a ordem e o encadeamento das fases (Comunicação, Planejamento, Modelagem, Construção, Entrega) em modelos sequenciais (e.g., Cascata), incrementais (e.g., Iterativo Incremental) e evolutivos (e.g., Evolucionário).
- **Justificar a escolha do modelo:** Discutir e defender a adequação de cada modelo de processo em diferentes cenários de projeto de software, considerando variáveis como clareza de requisitos, recursos disponíveis, tolerância a riscos e necessidade de *feedback* rápido, entre outras, que influenciam a escolha por um ou outro modelo [3] [10].
- **Identificar *trade-offs*:** Analisar as consequências da aplicação de cada modelo, reconhecendo *trade-offs* cruciais como tempo de entrega de valor ao cliente, uso eficiente de recursos, flexibilidade para acomodar mudanças e risco de retrabalho em fases tardias.

2.2 A Dinâmica Instrucional

A dinâmica "Encenando Modelos de Processo de Software" utiliza um kit de montagem (peças de madeira coloridas e cartas-modelo com imagens diversas – Figura 1) para simular o desenvolvimento de um produto de software. O desafio consiste em reproduzir

5 imagens (as "funcionalidades" do produto) seguindo, sucessivamente, as restrições de fluxo de um modelo sequencial, um incremental e um evolutivo.

A escolha por uma dinâmica ativa está fundamentada na Aprendizagem Experiencial, popularizada pelo ciclo de Kolb [11], que enfatiza a importância de "aprender fazendo". Essa abordagem segue uma sequência em que o aluno primeiro vive a experiência (executa a dinâmica), depois reflete sobre ela (discute sobre o que ocorreu), para então conceituar o aprendizado (relacionando com a teoria de ES) e, por fim, aplicá-lo em novos contextos. A seguir, detalhamos os papéis, materiais e o roteiro apresentado para a realização da dinâmica.

2.2.1 Papéis e Responsabilidades

A dinâmica utiliza o *role-play* como ferramenta para concretizar as responsabilidades dos participantes e os pontos de transferência de trabalho entre eles, mimetizando as funções essenciais de uma equipe de desenvolvimento de software como mostra o Quadro 1.

Quadro 1. Papéis e responsabilidades assumidas

Papel	*Responsabilidade Primária*
Gerente	Gerencia tempo, recursos, atividade e delegação. Decide a ordem das "funcionalidades" (cartas) e é o ponto de contato com o "Cliente".
Analista de Requisitos	Coleta, interpreta e traduz as necessidades (cartas) para a equipe de desenvolvimento. Separa as peças (artefatos). Realiza negociações para possíveis concessões.
Desenvolvedor	Constrói o "produto" (monta as imagens) com base nas especificações (cartas e peças entregues pelo Analista).
Testador	Avalia se o produto montado (imagem) está de acordo com a especificação original (carta) e registra não conformidades (bugs).
Cliente	(Papel assumido pela professora) Expressa as necessidades e fornece feedback nas entregas.

2.2.2 Materiais Utilizados

Os materiais foram selecionados para criar uma analogia tangível com os artefatos e ferramentas utilizados no desenvolvimento de software:

- Cartas com Imagens (Requisitos e Especificação Funcional): Imagens-modelo que detalham o produto esperado. Servem como a fonte primária de informação para o Analista e o oráculo para o Testador.
- Kit de Montagem ("Código-Fonte"/Recursos de Construção): Conjunto de peças geométricas coloridas e variadas, utilizado pelos Desenvolvedores para construir o produto.
- Projetor e Quadro Branco: O projetor foi usado para a explicação inicial e o quadro branco para registrar demandas do Cliente e concessões/mudanças de escopo durante as rodadas.

- Mesas (Ambiente de Desenvolvimento/Infraestrutura): Apoio físico essencial para a separação de peças, a montagem das imagens e a organização dos artefatos em lotes ou incrementos.

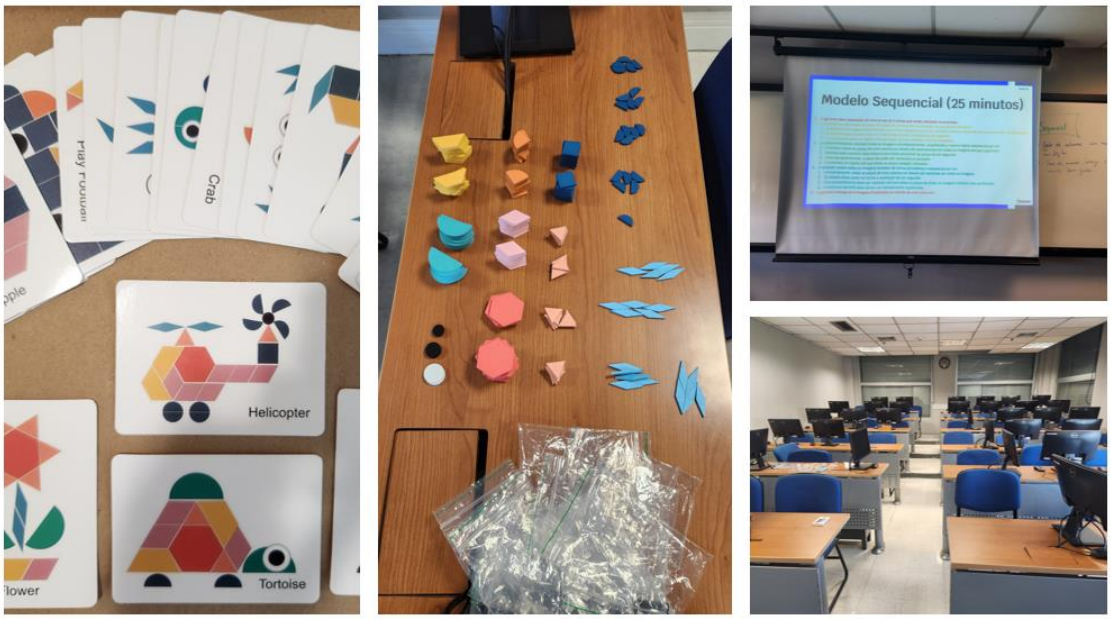

Figura 1. Exemplo de recursos disponíveis para a dinâmica

2.2.3 Fluxo da Dinâmica por Modelo de Processo de Software

A dinâmica é dividida em três rodadas, cada uma aplicando as regras específicas de um modelo de processo. Em todas as rodadas foram apresentadas as demandas da cliente e a necessidade de montagem de 5 imagens. O tempo estipulado visa intensificar as restrições e *trade-offs* de cada fluxo.

Rodada 1: Modelo Sequencial - 25 minutos

A rodada é caracterizada pela sequência linear, pela entrega em lote ao final e pela dependência entre as fases.

- Planejamento (Gerente): O Gerente define o escopo selecionando todas as 5 cartas de uma única vez com base nas demandas fornecidas pelo cliente.
- Análise (Analista): O Analista de Requisitos separa todas as peças necessárias para as 5 cartas. Esta separação deve seguir a restrição de ordem sequencial por cor (finalizar todas as peças de uma cor antes de passar para a próxima).
- Construção (Desenvolvedores): Os Desenvolvedores montam todas as 5 imagens simultaneamente, respeitando a mesma sequência rígida de cor em cor imposta pelo Analista. Uma peça, uma vez posicionada, não pode ser removida ou ajustada (alto custo de mudança).
- Testes e Entrega (Testador/Gerente): O Testador avalia todas as 5 imagens de uma só vez, apenas após a conclusão da montagem (atuação tardia). O Gerente entrega as 5 imagens finalizadas ao Cliente em um único momento.

O objetivo é simular a ociosidade das equipes na espera pelo lote e o alto risco de retrabalho tardio, já que o erro só é detectado no final do processo.

Rodada 2: Modelo Incremental - 15 minutos

A rodada é caracterizada por ciclos curtos que resultam em entregas operacionais por estágio.

- Planejamento (Gerente): O Gerente escolhe apenas uma carta por vez (um incremento) com base nas demandas fornecidas pelo cliente, permitindo que o processo comece imediatamente sem esperar pela definição de todo o escopo.
- Análise (Analista): O Analista seleciona apenas as peças necessárias para a carta escolhida no incremento atual. Após a conclusão da separação das peças (mantendo a ordem sequencial por cor, mas limitada àquela carta), a carta e as peças são imediatamente entregues aos Desenvolvedores.
- Construção (Desenvolvedores): Os Desenvolvedores montam a imagem completa referente àquela única carta, seguindo a ordem de cores. A montagem resulta em uma imagem finalizada e utilizável ao final do ciclo.
- Testes e Entrega (Testador/Gerente): O Testador avalia a imagem finalizada neste incremento. O Gerente entrega a imagem operacional ao Cliente, obtendo feedback imediato.
- Iteração: Depois que a primeira imagem é finalizada e validada, o processo recomeça do passo 1 para a próxima carta. O ciclo se repete até que todas as imagens estejam prontas.

O objetivo é simular o fluxo de trabalho contínuo e a entrega antecipada de valor, facilitando o *feedback* rápido e reduzindo o risco de erros se propagarem para todo o sistema.

Rodada 3: Modelo Evolutivo - 20 minutos

A rodada é caracterizada pelo desenvolvimento em versões parciais de todo o sistema e pelo ciclo contínuo de *feedback* para refinar os requisitos.

- Planejamento (Gerente): O Gerente seleciona todas as 5 cartas de uma só vez com base nas demandas fornecidas pelo cliente. O objetivo, no entanto, é evoluir a montagem em versões sucessivas e não finalizar tudo de imediato.
- Análise (Analista): Em cada um dos cerca de cinco ciclos, o Analista escolhe e entrega apenas uma porcentagem aproximada (e.g., 20%) das peças para todas as 5 cartas simultaneamente. A cada iteração, mais 20% são adicionados.
- Construção (Desenvolvedores): Os Desenvolvedores montam versões parciais das 5 imagens a cada ciclo (Iteração 1: 20%; Iteração 2: +20%, etc.). Eles precisam fazer suposições sobre o posicionamento das peças futuras. Uma peça, uma vez posicionada, não pode ser removida (simulando que erros de arquitetura se propagam).

- Testes e Feedback (Testador/Gerente): O Testador avalia as versões intermediárias a cada iteração, verificando o conjunto expandido (20%, 40%...). O Gerente apresenta essas versões parciais ao Cliente para obter feedback constante. É possível que o Cliente aceite algumas imagens como entregues, mesmo que incompletas.
- Iteração: O ciclo análise → construção → teste → feedback se repete cerca de cinco vezes, adicionando 20% das peças a cada vez, até que as imagens estejam completas.

O objetivo é simular o risco de arquitetura, a incerteza sobre o requisito final e a necessidade de adaptação contínua com base no *feedback* de protótipos.

2.2.4 Linha do Tempo da Sessão

A aplicação da dinâmica foi estruturada em 100 minutos (dois tempos de aula), utilizando uma metodologia de ciclos de execução e reflexão para cada modelo. Após uma introdução e *setup* de 10 minutos para a atribuição de papéis, o tempo restante foi dedicado a três rodadas centrais: Sequencial (3 min de explicação + 25 min de execução + 5 min de discussão); Incremental (3 min de explicação + 15 min de execução + 5 min de discussão); e Evolutivo (3 min de explicação + 25 min de execução + 5 min de discussão).

3. Implementação em Sala e Análise dos Achados

A dinâmica foi aplicada no dia 22/08/2025 em uma turma de 5 alunos, conforme detalhado na Seção 2.1. O *setup* inicial de papéis distribuiu as funções (1 Gerente, 1 Analista de Requisitos, 2 Desenvolvedores e 1 Testador), sendo que a professora assumiu o papel de Cliente.

3.1 Rodada 1: Modelo Sequencial (Cascata)

O Modelo Sequencial foi aplicado seguindo as regras de fluxo linear apresentadas. Antes da execução, a seguinte informação foi fornecida ao grupo: “Gosto de números com mais de um dígito”; “Tons de amarelo, laranja e rosa vão muito bem juntos.”

3.1.1 Ociosidade, Gargalos e Falha na Avaliação de Viabilidade

O achado mais imediato na execução do Modelo Sequencial foi a ociosidade de recursos humanos. Durante os primeiros 10 minutos, o Analista de Requisitos trabalhou intensamente para mapear as peças das 5 cartas. Neste período, os Desenvolvedores e o Testador ficaram completamente inativos (ociosos), aguardando a entrega do grande lote de trabalho do Analista.

O principal erro, no entanto, ocorreu na fase de Planejamento: o Gerente não avaliou os recursos disponíveis antes de selecionar as cartas, violando a premissa de viabilidade técnica. Essa falha levou à escolha de cartas que não podiam ser montadas integralmente com o kit de peças, uma vez que a construção de todas as peças juntas exigia uma quantidade maior de peças de determinadas cores do que a disponível (restrição técnica, ou seja, falta de recursos).

O problema só foi detectado pelo Analista no meio da sua tarefa de separação de peças, quando este percebeu que as especificações não eram exequíveis com o "estoque" disponível.

Analogia com ES: A experiência em sala confirmou o risco clássico do Modelo Cascata [12] que é o de ter equipes paradas aguardando outra equipe terminar sua parte, gerando ociosidade e uso ineficiente do capital humano. Além disso, ao não validar corretamente os recursos, o Gerente acabou exemplificando a definição de escopo sem verificar restrições técnicas, o que é uma causa comum de insucesso em projetos sequenciais.

3.1.2 Retrabalho e Estouro de Prazo

A descoberta tardia da inviabilidade do escopo resultou em um retrabalho tanto de planejamento quanto de análise, confirmando a rigidez do modelo. O Analista de Requisitos teve que interromper sua tarefa e pedir troca de cartas ao Gerente. Foi então que o Gerente e o Analista organizaram melhor os recursos para poderem identificar mais facilmente a quantidade de peças disponíveis de cada cor e, assim, saber quando determinadas cartas poderiam ser montadas em conjunto ou não. O Gerente então fez uma nova seleção de cartas para montagem, refazendo tanto a fase de Planejamento quanto a de Análise que teve que começar do zero a separação de peças já iniciada – ver Figura 2.

Figura 2. Fluxo Sequencial

A consequência cumulativa da ociosidade inicial somada ao retrabalho forçou a docente a estender o tempo inicialmente estipulado como uma forma de tentar avançar nas fases de desenvolvimento. Mesmo com a extensão, o processo demorou muito mais do que o previsto e o trabalho acabou sendo abandonado sem sequer iniciar a fase de montagem das imagens uma vez que houve uma percepção de que os alunos estavam perdendo a motivação na dinâmica.

Analogia com ES: Este evento simulou o "Retrabalho caro em cascata": erros de escopo descobertos tarde exigem refazer várias fases completas, atrasando todo o cronograma e, em muitos casos, levando à interrupção do processo por prazo estourado, onde o projeto é abandonado sem uma entrega final ao cliente. A natureza sequencial e a entrega tardia de valor impediram a detecção precoce do erro de viabilidade.

3.2 Rodada 2: Modelo Incremental (Iterativo Incremental)

O Modelo Incremental foi executado em miniciclos para cada uma das 5 cartas, focando em entregar uma imagem completa por vez, o que transformou o fluxo de trabalho da

equipe. Antes da execução, a seguinte informação foi fornecida ao grupo: “Adoro a sensação de voar e nadar”; “Tenho verdadeira paixão por animais.”

3.2.1 Entrega Antecipada de Valor, Avaliação de Viabilidade Contínua e Possibilidade de Negociação

Nesta rodada, o tempo de ociosidade dos papéis foi drasticamente reduzido, pois a equipe trabalhou em paralelo a cada incremento, otimizando o uso de todos os recursos humanos. A entrega de cada incremento (uma imagem completa e operacional) era imediata, resultando em maior engajamento dos alunos.

A cada novo ciclo, o Gerente e o Analista podiam renegociar a prioridade e o escopo da próxima carta, demonstrando o planejamento adaptativo na prática. Além disso, antes de iniciar o novo incremento, os alunos realizavam uma avaliação dos recursos, verificando a viabilidade técnica para a próxima carta. Essa verificação evitou o problema de viabilidade que inviabilizou o Modelo Sequencial, pois ou possibilitou a escolha de uma nova carta compatível com os requisitos e os recursos ou possibilitou a realização de negociação com a Cliente em relação a aceitação de uma entrega adaptada (exemplo: “Turkey”) e/ou incompleta (exemplo: “Fish”) diante dos requisitos e recursos disponíveis – ver Figura 3.

Analogia com ES: O principal ganho foi a ilustração de que o cliente tem acesso a uma versão útil do software (com as funcionalidades prioritárias) em um tempo menor. A equipe funcionou como uma célula ágil, onde a equipe trabalha em paralelo, e a avaliação constante dos recursos (viabilidade técnica a cada entrega) garante que o projeto não colida com restrições orçamentárias ou de capacidade tardiamente.

3.2.2 Gestão de Múltiplos Ciclos e Estabilidade de Requisitos

A rodada incremental, embora mais eficiente, revelou a necessidade de coordenação entre múltiplos ciclos principalmente para garantir a viabilidade das entregas. O ambiente se mostrou um ponto fraco em uma dinâmica mais intensa em interação entre os membros, já que o espaço físico não era adequado para a dinâmica.

Por outro lado, como o requisito (a imagem na carta) era bem estabelecido e os recursos eram verificados a cada ciclo, estando disponíveis para os desenvolvedores antes do início da montagem de cada peça especificamente, os Desenvolvedores puderam focar na construção rápida da solução com menos decisões a tomar e posicionando as peças de maneira mais coerente ainda que seguindo uma ordem de posicionamento por cor. O feedback obtido do Cliente a cada entrega permitiu que a equipe ajustasse as prioridades, aumentando o alinhamento e a flexibilidade do projeto.

Figura 3. Fluxo Incremental

Analogia com ES: O modelo reforçou que, em incrementos com requisitos estáveis, o time foca em construção rápida. A necessidade de coordenação de múltiplos ciclos, no entanto, demanda maior maturidade gerencial do Gerente, mas é compensada pela entrega antecipada de valor funcional e pelo feedback que influencia a próxima escolha.

3.3 Rodada 3: Modelo Evolutivo (Evolucionário)

Nesta rodada, o foco foi no desenvolvimento das 5 imagens em paralelo, mas de forma evolutiva, adicionando apenas cerca de 20% das peças em cada um dos cinco ciclos. O objetivo era refinar a solução (as 5 imagens) gradualmente em versões sucessivas, demandando adaptação e gestão dinâmica de prioridades e escopo, o que tornou o processo mais dinâmico em comparação com o Sequencial. Antes da execução, a seguinte informação foi fornecida ao grupo: “Só vale sobreposição.”

3.3.1 Incertezas e Risco de Problemas Arquiteturais

A principal dificuldade percebida pelos alunos foi a incerteza inerente à construção gradual. O Analista acabou não focando na escolha de peças de forma a garantir uma montagem mais lógica de cada imagem. Isso quer dizer que a sobreposição exigida nas peças nem sempre ocorreu da forma como descrita nos requisitos, já que as peças foram entregues “fora de ordem”.

Além disso, os Desenvolvedores tiveram que lidar com a incerteza sobre as peças futuras, pois precisavam adivinhar onde as peças futuras iriam se encaixar (suposições de localização – ver montagem intermediária da Figura 4), embora tivessem a especificação completa. Isso ilustrou que projetos evolutivos exigem suposições arquiteturais que podem não se confirmar nas versões finais.

A consequência prática desse risco foi o retrabalho: as peças posicionadas nas iterações iniciais (20% e 40%) em alguns casos impediram a colocação correta das peças entregues nos ciclos subsequentes (exemplo: Cock), exigindo que os Desenvolvedores realocassem partes já montadas ou ainda entrassem em acordo com o Cliente sobre o que poderia ser entregue.

Analogia com ES: O planejamento iterativo sofre com requisitos mal definidos. A incerteza traduziu-se no risco de que decisões ruins de iterações iniciais não fossem corrigidas, podendo resultar em um produto mal estruturado ou com baixa qualidade

estética, ilustrando a acumulação de dívida técnica quando há bugs ou erros arquiteturais nos protótipos iniciais.

3.3.2 Evolução da Solução e Desafios da Prototipação

A distinção crucial entre o modelo Evolutivo e o Incremental tornou-se evidente na natureza das entregas. As versões intermediárias (20%, 40%, 60% e 80%) eram entregues pouco funcionais – as imagens estavam incompletas e não eram "utilizáveis", dificultando a avaliação de valor pelo Cliente – simulando que protótipos evolutivos são pouco funcionais no início e podem frustrar stakeholders.

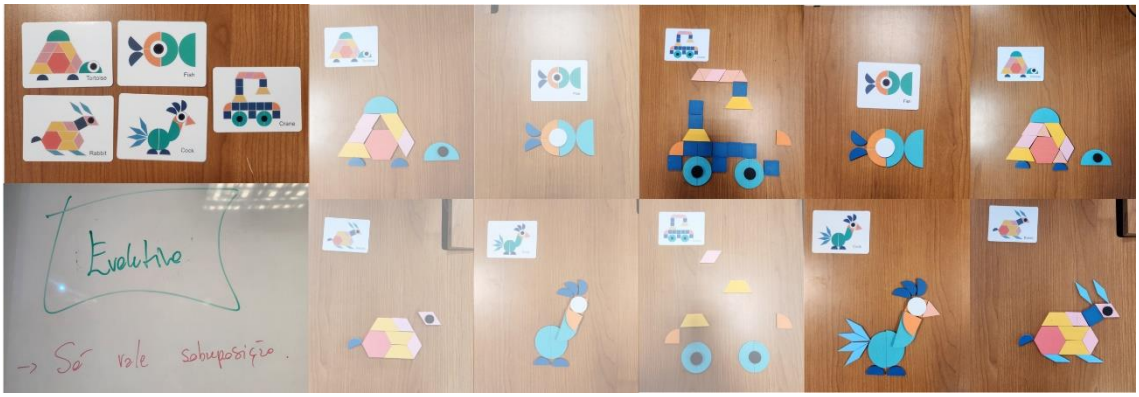

Figura 4. Fluxo Evolutivo

Apesar disso, a iteração contínua permitiu a evolução da solução. O Gerente apresentou a versão de 80% das imagens para o Cliente, que aceitou uma como "suficientemente boa" para a entrega – até porque os recursos de um tipo de peça tinham sido esgotados, enquanto outras exigiram um ciclo adicional. Essa capacidade do Cliente ajustar prioridades ou aceitar a solução em um estado incompleto validou o modelo. Contudo, a necessidade de montar as imagens em várias versões parciais indicou que o desenvolvimento pode ser mais demorado no tempo total, pois os ciclos de prototipação aumentam o esforço. Além disso, a simulação de possível falta de recursos no meio do processo reforçou o risco de não se ter orçamento, tempo ou tecnologia adequada para a entrega final, característica de projetos evolutivos longos.

A seção seguinte aprofundará as implicações pedagógicas desses achados, contrastando as experiências Sequencial, Incremental e Evolutiva.

4. Análise Reflexiva da Docente e Discussão Pedagógica

Embora a dinâmica tenha se mostrado altamente eficaz em atingir os Objetivos de Aprendizagem de nível analítico (diferenciação de fluxos e identificação de *trade-offs*), a sua aplicação prática gerou importantes reflexões sobre o equilíbrio entre o rigor pedagógico, o engajamento lúdico e as limitações do ambiente físico.

4.1 O Desafio do Engajamento e a Questão da Ociosidade

O impacto do Modelo Sequencial no engajamento dos participantes constituiu o principal ponto de preocupação da docente. Embora o rigor em simular a ineficiência inerente aos processos sequenciais fosse didaticamente essencial, ele resultou na ociosidade forçada dos Desenvolvedores e do Testador. Essa inatividade, crucial para evidenciar o *trade-off* de tempo e desperdício de recursos, tornou a dinâmica mais educativa do que lúdica para os participantes na primeira rodada.

O tempo inicialmente alocado de 25 minutos para o Modelo Sequencial se mostrou insuficiente. Foi necessário intervir, estendendo e, em seguida, encurtando a rodada, pois os alunos ociosos demonstraram desinteresse visível. O dilema pedagógico se estabeleceu: a inatividade simulada é um achado crucial para a internalização do problema do Modelo Sequencial, mas pode comprometer a natureza ativa da dinâmica. A decisão de encurtar a execução foi baseada na constatação de que a lógica da ociosidade, do retrabalho e da insatisfação do cliente já havia sido bem internalizada pelos alunos presentes, mesmo sem a conclusão integral do trabalho.

4.2 Adaptação de Tempo e Velocidade de Internalização

A flexibilidade na gestão do tempo alocado revelou diferenças críticas entre os modelos, embora seja natural que a primeira rodada fosse marcada por um tempo maior até que os alunos internalizassem completamente o fluxo, os papéis e as regras da dinâmica.

O Modelo Sequencial, por ser o primeiro, sofreu tanto com o entendimento inicial da dinâmica quanto com a imaturidade gerencial da equipe. O Gerente não se atentou à necessidade de avaliar os recursos antes de escolher as 5 cartas. Essa falha de planejamento levou ao erro de escopo (escolha de cartas inviáveis), que se traduziu em retrabalho para o Analista. Assim, a lentidão e a crise inicial da rodada com o Cascata não podem ser integralmente atribuídas ao modelo de processo em si, mas sim à curva de aprendizado da dinâmica e, claro, à falha na avaliação de viabilidade do software que pode ocorrer com outros modelos (e ocorreu, porém, com menor impacto).

Os dois modelos seguintes tiveram o proveito desse entendimento inicial da dinâmica e se tornaram naturalmente mais dinâmicos. No Modelo Incremental, a segmentação do problema em incrementos menores permitiu uma execução fluida. O fluxo reduziu a ociosidade, tornando o trabalho mais ágil e culminando na entrega de valor funcional em estágios. O Modelo Evolutivo, embora mantivesse o dinamismo do Incremental, introduziu o desafio do risco de arquitetura e retrabalho. Os Desenvolvedores, ao montar as versões parciais, geraram retrabalhos de montagem devido à incerteza sobre peças futuras, exigindo concessões pelo Cliente nas entregas.

Para não sobrecarregar a Rodada Sequencial com o aprendizado inicial da dinâmica, o ideal em futuras aplicações seria iniciar com uma rodada rápida de prática. Essa rodada inicial, sem avaliação de conteúdo, permitiria que os alunos compreendessem plenamente os fluxos e as regras antes de aplicar o conhecimento aos modelos de processo.

4.3 Impacto do Ambiente Físico na Execução da Dinâmica

A dinâmica foi executada em um laboratório de informática, um ambiente adaptado para o exercício, o que influenciou negativamente a organização e o fluxo do trabalho. As mesas fixas, projetadas para acomodar computadores e monitores, forçaram o afastamento das máquinas e a reorganização improvisada do espaço para a montagem e a movimentação das peças – ver detalhes na Figura 1. Essa limitação física gerou os seguintes problemas:

- Prejuízo à Colaboração e Fluxo: A organização das peças e cartas precisou ser compactada. Em modelos como o Sequencial, onde o Analista atuou com grandes quantidades de peças, a falta de espaço amplo e dedicado para espalhar e inspecionar as peças dificultou a organização e aumentou a probabilidade de erros.

- Dificuldade de Visualização: A necessidade de afastar os equipamentos e usar as pequenas mesas do laboratório como superfície de trabalho prejudicou a visualização clara das 5 imagens sendo montadas em paralelo, o que é especialmente crucial na rodada Evolutiva de forma a reduzir retrabalho de escolha de peças erradas para iniciar.

Acredita-se que uma sala sem equipamentos fixos (uma sala de aula tradicional ou sala de *design thinking*) seria mais adequada, pois permitiria uma disposição circular ou em "U" das mesas, facilitando a interação e a separação clara dos "artefatos" de cada papel.

5. Limitações e Considerações sobre Replicações

A aplicação da dinâmica foi concebida primariamente como uma intervenção pedagógica de Aprendizagem Ativa, e não como um estudo de pesquisa formal. Dessa natureza derivam importantes limitações metodológicas que devem ser reconhecidas.

5.1 Limitações Metodológicas

A principal limitação deste relato de experiência reside na impossibilidade de garantir que os benefícios observados (maior clareza conceitual, internalização de *trade-offs* e diferenciação entre modelos) sejam fruto exclusivo da dinâmica.

Validade Interna (Fator Confundidor): A dinâmica foi aplicada em um momento específico do curso (terceiro semestre) e inserida em um ambiente temático de *back-end* com outras disciplinas correlatas. É possível que o aprendizado tenha sido influenciado pela sinergia com os demais conteúdos ou mesmo com conteúdos já vistos em semestres anteriores. Além disso, a execução da dinâmica foi precedida por aulas expositivas teóricas sobre os modelos (Cascata, Incremental, Evolutivo), tornando difícil isolar o quanto o aprendizado é atribuível à aula teórica e o quanto é atribuível à vivência prática.

Amostra Reduzida: A dinâmica envolveu uma turma pequena, composta por apenas 5 alunos presentes. Essa amostra não é representativa da população estudantil de Engenharia de Software e, portanto, os achados e as conclusões da docente não podem ser generalizados para turmas maiores ou para diferentes perfis de alunos (por exemplo, alunos já estagiando).

Coleta de Dados Subjetiva: O artigo baseia-se em um Relato de Experiência da docente baseado em observação e discussões realizadas com os alunos em cada uma das rodadas, e não em uma coleta de dados sistematizada (e.g.: pré-testes, pós-testes, entrevistas semiestruturadas ou análise estatística). A percepção de sucesso do aprendizado é, portanto, qualitativa e subjetiva, focada na observação do engajamento e nas respostas dadas pelos alunos durante a seção de discussão.

5.2 Abordagem de Avaliação dos Alunos com a Atividade

Para mitigar qualquer constrangimento ou pressão de desempenho que pudesse afetar o aprendizado e a participação, foram adotadas diretrizes claras em relação à avaliação e ao objetivo da atividade.

A dinâmica foi classificada como uma das oito avaliações complementares dos alunos envolvidos, que somavam 20% da nota final da disciplina. Foi comunicado aos alunos que a simples participação no exercício contabilizaria a nota integral da atividade, independentemente da corretude da execução ou da entrega final do produto (as imagens montadas).

Essa abordagem desvinculou a nota do desempenho e da perfeição da execução. O objetivo era "forçar" os alunos a aprenderem fazendo, a cometerem os erros típicos de cada modelo (ociosidade, retrabalho) sem o medo de serem punidos por isso. Essa medida ética garantiu que os alunos se sentissem à vontade para assumir riscos nos papéis (especialmente Gerente e Analista), maximizando a vivência dos *trade-offs* de cada modelo.

A observação e o registro dos achados (ociosidade, frustração, retrabalho) foram utilizados exclusivamente para fins de reflexão pedagógica e estão sendo documentados neste relato de experiência. Nenhum aluno foi identificado individualmente, preservando a sua privacidade.

5.3 Pontos de Atenção para Replicação

Em uma análise retrospectiva, apesar da eficácia percebida em traduzir conceitos abstratos em experiências concretas, o docente pondera sobre a replicação da dinâmica no formato atual devendo-se atentar para o risco de ociosidade que é a principal barreira para a diversão e pode ser desmotivadora; e o fato de que o ambiente para execução da dinâmica tem forte relação com a fluidez da atividade, minimizando fatores externos que influenciam negativamente a organização e o foco do grupo.

6. Considerações Finais

O objetivo deste artigo foi relatar a experiência da aplicação de uma dinâmica de montagem de imagens para simular a execução de modelos de processos de software (Sequencial, Incremental e Evolutivo) com alunos de graduação em uma disciplina de Engenharia de Software.

O relato demonstrou que a dinâmica se constituiu em uma ferramenta de Aprendizagem Ativa eficaz para transpor a barreira da abstração conceitual. A vivência dos diferentes fluxos permitiu aos alunos diferenciar, na prática, as consequências de cada modelo: o Modelo Sequencial (Cascata) expôs de forma vívida e indesejada os gargalos, a ociosidade da equipe, o acúmulo de retrabalho e o risco de entrega tardia; o Modelo Incremental evidenciou a fluidez do trabalho em paralelo, a redução do tempo de *feedback* e a entrega de valor mais rápida por meio de estágios operacionais; e o Modelo Evolutivo ilustrou a necessidade de gestão da incerteza e o risco de dívida técnica acumulada pelas suposições arquiteturais iniciais.

Conclui-se que a dinâmica é uma ferramenta de validação de conceito, mas exige uma gestão de tempo ainda mais estrita por parte do docente, principalmente na fase Sequencial, e um planejamento cuidadoso do ambiente para garantir que a discussão seja sustentada pela experiência negativa (ociosidade/retrabalho) sem comprometer o engajamento geral. A principal contribuição pedagógica reside em permitir que os alunos vivenciem o "porquê" das metodologias iterativas terem surgido como resposta aos problemas dos modelos sequenciais, em vez de apenas decorarem suas definições teóricas.

Em termos de futuras melhorias, propõe-se a realização de um estudo comparativo com grupos de controle para validar estatisticamente a influência da dinâmica na retenção do aprendizado. Outras vertentes de trabalho incluem a adaptação da dinâmica para simular modelos de processo Ágeis (como XP e Scrum), introduzindo conceitos como *sprint review* e *daily stand-ups*.

Referências

[1] M. R. d. A. Souza, L. Veado, R. T. Moreira, E. Figueiredo e H. Costa, "A Systematic Mapping Study on Game-Related Methods for Software Engineering Education," *Information and Software Technology,* vol. 95, pp. 201-218, march 2018.

[2] W. Lemos, J. Cunha e J. Saraiva, "Ensino de Engenharia de Software em um Curso de Sistemas de Informação: Uma Análise dos Problemas e Soluções na Perspectiva de Professores e Alunos," em *Anais do XXVII Workshop sobre Educação em Computação - XXXIX Congresso da Sociedade Brasileira de Computação*, Belém, 2019.

[3] R. S. Pressman, Software Engineering - A Practitioner's Approach, 7th ed., New York: McGraw-Hill Professional, 2010.

[4] Y. S. V. P. J. M. a. G. H. Dieter Landes, "Learning and Teaching Software Process Models," em *Proceedings of the 2012 IEEE Global Engineering Education Conference*, Marrakech, 2012.

[5] S. Freeman, S. L. Eddy, M. McDonough, M. K. Smith, N. Okoroafor, H. Jordt e M. P. Wenderoth, "Active Learning Increases Student Performance in Science, Engineering, and Mathematics," *Proceedings of the National Academy of Sciences,* vol. 111, nº 23, pp. 8410-8415, June 2014.

[6] C. França, J. A. Cunha, D. Adjarde e F. Alan, "Uma Investigação sobre Estilos de Aprendizagem e Hábitos de Estudo de Engenheiros de Software," em *Anais do IX Fórum de Educação em Engenharia de Software - XXX Simpósio Brasileiro de Engenharia de Software*, Maringá, 2016.

[7] R. Prikladnicki, A. B. Albuquerque, C. G. v. Wangenheim e R. Cabral, "Ensino de Engenharia de Software: Desafios, Estratégias de Ensino e Lições Aprendidas," em *Anais do Fórum de Educação em Engenharia de Software - XXIII Simpósio Brasileiro de Engenharia de Software*, Fortaleza, 2009.

[8] V. Moura e G. Santos, "ProcSoft: A Board Game to Teach Software Processes Based on ISO/IEC 29110 Standard," em *Proceedings of the XVII Brazilian Symposium on Software Quality*, Curitiba, 2018.

[9] K. Gama, "An Experience Report on Using LEGO-based Activities in a Software Engineering Course," em *Proceedings of the XXXIII Brazilian Symposium on Software Engineering*, Salvador, 2019.

[10] I. Sommerville, Engenharia de Software, 10ª ed., São Paulo: Pearson Education do Brasil, 2018, pp. 29-55.

[11] D. A. Kolb, “The Process of Experiential Learning,” em *Experiential Learning: Experience as the Source of Learning and Development*, Englewood Cliffs, Prentice Hall, 1984, pp. 20-38.

[12] W. W. Royce, “Managing the Development of Large Software Systems,” em *Proceedings of the IEEE WESCON*, Los Angeles, 1970.