APRIMORAMENTO SISTEMÁTICO DO PBL NA ENGENHARIA DE SOFTWARE: UM MÉTODO BASEADO EM OBJETIVOS DE APRENDIZAGEM E VISÕES ARQUITETURAIS

1 INTRODUÇÃO

O ensino superior enfrenta desafios crescentes para preparar profissionais capazes de lidar com a complexidade e a rápida evolução do mercado de trabalho, especialmente nas áreas de tecnologia e engenharia. Nesse contexto, o Project-Based Learning (PBL) emerge como uma metodologia promissora, capaz de proporcionar uma formação mais abrangente e alinhada às demandas contemporâneas (Barrows, 1996).

Neste cenário de inovação educacional, destaca-se o Instituto de Tecnologia e Liderança (Inteli), uma instituição de ensino superior pioneira no Brasil na adoção integral do modelo PBL.

A estrutura curricular do Inteli é organizada em módulos de aprendizagem chamados LBLs (Learning BackLogs), que são constantemente revisados e ajustados para garantir a relevância e eficácia do ensino. Estes módulos são projetados para integrar conhecimentos técnicos de computação, matemática e física com competências de liderança, negócios e design de experiência do usuário, proporcionando uma formação holística aos estudantes.
Um aspecto fundamental dessa estrutura é a forma como as Diretrizes Curriculares Nacionais (DCNs) são incorporadas e distribuídas ao longo dos LBLs. As disciplinas tradicionais, conforme definidas nas DCNs, são cuidadosamente decompostas em pacotes de assuntos que são então permeados e revisitados pelos diferentes LBLs. O diferencial dessa abordagem está na forma como os assuntos são apresentados: eles reaparecem em diversos momentos do curso, sendo progressivamente aprofundados ou examinados sob diferentes perspectivas a cada iteração. Essa estratégia de revisitação e aprofundamento gradual permite que os estudantes construam um entendimento mais sólido e multifacetado dos temas. Desta forma, o Inteli consegue manter total aderência às exigências legais e às disciplinas definidas nas DCNs, enquanto oferece uma experiência de aprendizagem mais fluida e conectada.

Porém, o fato é que esta abordagem requer um planejamento integrado de todas as vertentes e uma revisão constante para assegurar que todos os conteúdos essenciais sejam cobertos de maneira adequada, mantendo ao mesmo tempo a flexibilidade necessária para adaptar-se às demandas em constante evolução do mercado de tecnologia.

Neste contexto específico do Inteli, mas com implicações para a implementação do PBL no ensino superior em geral, emergem questões importantes sobre a implementação e aprimoramento desta metodologia:

- Como realizar a evolução sistemática dos módulos de aprendizagem baseados em projetos (LBLs)?
- Quais aspectos da aprendizagem devem ser utilizados como base para compor os requisitos de revisão?
- Como os pacotes de conceitos e práticas podem ser reconfigurados sem prejuízo das disciplinas essenciais definidas nas Diretrizes Curriculares Nacionais estabelecidas pelo MEC?

Portanto, o objetivo desta pesquisa é desenvolver um método de evolução contínua dos módulos de aprendizagem baseados em projetos (LBLs), visando compreender e aprimorar a jornada de aprendizado do aluno ao longo de um curso de graduação em

Engenharia de Software. Para isso, propõe-se uma abordagem que combina a taxonomia de Bloom com as dimensões arquiteturais de sistemas de software distribuídos, baseadas nas normas ISO/IEC 42010, IEEE 1470 e ISO/IEC 10746. Cada uma das visões propostas, baseadas nestas dimensões de arquitetura de sistemas, incorpora tanto habilidades técnicas quanto competências socioemocionais, essenciais para a formação completa do profissional de Engenharia de Software.

Como contribuição, a criação de um instrumento sistemático que permite observar e medir de forma pragmática o processo de aprendizagem dentro dos módulos e entre eles. Este instrumento utiliza métricas baseadas nas visões de arquitetura mencionadas, correlacionando-as com os níveis da taxonomia de Bloom. Isso nos permite avaliar não apenas o domínio técnico dos alunos, mas também sua capacidade de aplicar, analisar, avaliar e criar soluções em contextos complexos de sistemas distribuídos.

Além disso, a abordagem visa estabelecer requisitos claros para o aprimoramento do aprendizado, comparando o estágio atual dos módulos com o estágio desejado. Isso permite identificar lacunas e oportunidades de melhoria, sempre alinhadas com as demandas do mercado e com as melhores práticas da engenharia de software.

2 REVISÃO BIBLIOGRÁFICA SOBRE PBL

O Project-Based Learning (PBL) tem sido amplamente estudado e implementado em diversas instituições de ensino superior ao redor do mundo. Esta seção apresenta uma revisão da literatura sobre o uso do PBL no ensino superior, com ênfase em sua aplicação na área de Engenharia de Software.

Alguns trabalhos que estão relacionados com o PBL, evidenciam a oportunidade da proposta de sistematização deste artigo: O desafio da inovação e da melhoria contínua está bem enriquecido nas publicações como as indicadas nas publicações (Felder *et al*, 2004), (Habimorad, 2024), (Markham *et al*, 2008). As experimentações relacionadas à melhoria e eficiência dos projetos PBL na área de engenharia, os resultados obtidos e destaques de melhoria podem ser extraídas das publicações (Hayashi *et al*, 2021), (Hayashi *et al*, 2020), (Cugnasca *et al*, 2024) que desafiaram a pandemia para trazer a imersão virtual em experiências práticas de laboratório de circuitos digitais, usando IoT e placas de circuitos de hardware FPGA. Em (Hayashi *et al*, 2023) os *soft skills* e *hard skills* foram destacadas e medidas como aspectos importantes na aprendizagem do engenheiro. As publicações de (Apeanti *et al*, 2021), (Fioravanti *et al*, 2018) e (Zhang e Hu, 2024) permitiram um panorama sobre a busca continuada de avaliação de aprendizagem e melhorias relacionados, baseada em medidas de resultados e de eficácia das práticas aplicadas em atividades colaborativas.

3 ADOÇÃO DE PBL NO INTELI E PERSPECTIVAS DE APRENDIZAGEM

Este trabalho propõe uma nova abordagem para aprimorar a implementação do PBL em cursos de engenharia de software, utilizando duas estruturas conceituais complementares. Primeiramente, apresentamos o uso da taxonomia de Bloom como instrumento para projetar e implementar os Learning Backlogs (LBLs), considerando os níveis de aprendizado desejados. Em seguida, introduzimos uma adaptação das dimensões de aprendizado baseadas em visões arquiteturais de sistemas de software distribuídos, derivadas das normas ISO/IEC 42010 e ISO/IEC 10746. A combinação dessas abordagens visa fornecer um framework para o design, implementação e avaliação de módulos de

aprendizagem, proporcionando uma visão sistêmica da jornada de aprendizado dos alunos de engenharia de software.

3.1 Taxonomia de Bloom como instrumento de design e evolução de LBLs

A taxonomia de Bloom é utilizada nesta proposta como um instrumento para projetar, implementar e evoluir os Learning Backlogs (LBLs). Esta abordagem permite uma estruturação sistemática dos conteúdos, atividades e avaliações, garantindo coerência e integração dos mecanismos de ensino e aprendizagem ao longo do curso. Os seis níveis da taxonomia de Bloom aplicados são:

- Memorizar: Reconhecer e recordar informações relevantes;
- Compreender: Interpretar e explicar ideias ou conceitos;
- Aplicar: Usar informações em novas situações;
- Analisar: Distinguir entre as diferentes partes e examinar a estrutura;
- Avaliar: Justificar uma decisão ou curso de ação;
- Criar: Produzir trabalho original ou propor soluções alternativas.

Propomos o uso destes seis níveis no contexto dos LBLs, de acordo com o conteúdo apresentado na Tabela 1.

Tabela 1 - Proposta de uso da Taxonomia de Bloom

Memorizar	MMB	Básico	Recorda informações básicas, como fatos simples, definições ou datas importantes.
	MMI	Intermediário	Lembra-se de detalhes específicos e pode recitar informações com alguma precisão.
	MMA	Avançado	Recupera informações detalhadas sem auxílio e demonstra domínio completo dos fatos.
Compreender	COB	Básico	Explica conceitos em suas próprias palavras e consegue resumir informações essenciais.
	COI	Intermediário	Interpreta informações, traduz ideias para diferentes contextos e fornece exemplos adequados.
	COA	Avançado	Analisa implicações dos conceitos e consegue prever resultados baseados em sua compreensão.
Aplicar	APB	Básico	Utiliza conhecimentos adquiridos em situações familiares com orientação.
	API	Intermediário	Aplica conceitos em novos contextos com pouca ou nenhuma assistência.
	APA	Avançado	Resolve problemas complexos de forma autônoma, transferindo conhecimentos para situações inéditas.
Analisar	ANB	Básico	Identifica partes componentes de um todo e reconhece relações simples entre elas.
	ANI	Intermediário	Diferencia fatos de opiniões e examina padrões ou tendências nos dados.
	ANA	Avançado	Avalia estruturas complexas, identifica causas subjacentes e estabelece relações multifacetadas.
Avaliar	AVB	Básico	Expressa opiniões pessoais com justificativas básicas.
	AVI	Intermediário	Critica ideias ou trabalhos com base em critérios estabelecidos, oferecendo feedback construtivo.
	AVA	Avançado	Defende argumentos com base em evidências sólidas, compara diferentes perspectivas e propõe melhorias fundamentadas.
Criar	CRB	Básico	Combina informações conhecidas para formar ideias simples com orientação.
	CRI	Intermediário	Desenvolve soluções originais ou produtos em contextos conhecidos.
	CRA	Avançado	Concebe propostas inovadoras para problemas complexos, demonstrando alta criatividade e originalidade.

Fonte: Os autores.

Ao utilizar esta taxonomia, o time docente pode:

- Definir objetivos de aprendizagem claros para cada LBL;
- Selecionar materiais de estudo e atividades de instrução que correspondam ao nível cognitivo desejado;
- Desenvolver avaliações que reflitam adequadamente o nível de compreensão esperado;
- Definir artefatos de desenvolvimento de projeto aderentes ao nível cognitivo desejado;
- Garantir uma progressão adequada do conhecimento ao longo do curso.

Por exemplo, no módulo M5, que aborda o projeto de sistemas usando plataformas distribuídas (redes, internet e cloud computing), os objetivos de aprendizagem para Requisitos Funcionais (RF) e Requisitos Não Funcionais (RNF) são estabelecidos nos níveis "Compreender Básico" (COB) e "Aplicar Básico" (APB). Na prática, pode-se pensar em trabalhar da seguinte forma:

- COB - Instrução: Apresentar aos alunos diversos exemplos de RF e RNF em diferentes contextos de sistemas.
  - Atividade: Pedir aos alunos que classifiquem uma lista de requisitos misturados em RF ou RNF, justificando suas escolhas.
- APB - Instrução: Fornecer um estudo de caso de um sistema simples e demonstrar como identificar e documentar RF e RNF.
  - Atividade: Solicitar que os alunos, em grupos, identifiquem e documentem RF e RNF para um novo recurso a ser adicionado ao projeto que estão desenvolvendo no módulo.

Considerando ainda estes objetivos de aprendizado, não seria uma boa prática, por exemplo, pedir aos alunos que critiquem a arquitetura de um sistema complexo baseado em seus RF e RNF sem fornecer o conhecimento base necessário. Este objetivo já estaria relacionado ao objetivo “Avaliar” que é avançado para o módulos 5 e só deve aparecer em módulos subsequentes para estes mesmos assuntos.

Esta abordagem permite uma avaliação mais precisa e direcionada do progresso dos alunos, assegurando que eles desenvolvam habilidades cognitivas em todos os níveis, desde a memorização básica até a criação de soluções inovadoras.

3.2 Dimensões de aprendizado baseadas em visões arquiteturais de sistemas de software

Esta proposta se baseia em uma adaptação de normas internacionais reconhecidas na área de arquitetura de sistemas e software (Pressman, 2021). A ISO/IEC 10746, também conhecida como RM-ODP (Reference Model of Open Distributed Processing), fornece um *framework* para a especificação de sistemas distribuídos. A ISO/IEC 42010 estabelece um padrão para a descrição arquitetural de sistemas de software intensivos. Juntas, essas normas oferecem uma base sólida para a compreensão e a descrição de sistemas complexos em múltiplas perspectivas ou "visões".

Adaptando os princípios dessas normas para o contexto educacional, propomos cinco dimensões de aprendizado que integram tanto hard skills quanto soft skills, importantes para a formação de profissionais de engenharia de software. Esta abordagem reconhece que o mercado avalia precisamente o perfil dos estudantes considerando ambos os aspectos, preparando-os para os desafios multifacetados da indústria de software moderna:

- Visão de Negócio e Business Drivers (BD): Foca na arquitetura de um sistema distribuído alinhada com os direcionadores de negócio.
  - Hard skills: Assimilação dos fluxos críticos de negócios e desenvolvimento de uma arquitetura controlada e alinhada com os *business drivers*.

  - Soft skills: Capacidade de interagir, negociar e protagonizar aspectos digitais com pessoas não técnicas e em posições de alto poder decisório.
- Visão de Requisitos Funcionais (RF): Aborda os elementos de funcionalidade que usuários internos e externos (clientes) acessarão.
  - Hard skills: Aprendizado para conectar a digitalização do contexto de pessoas e produtos, exigindo implementações técnicas precisas e fluidas.
  - Soft skills: Negociação dos melhores fluxos de interação, aprimorando a usabilidade e adicionando valor percebido.
- Visão de Requisitos Não Funcionais (RNF): Foca nos pilares que sustentam a plataforma, incluindo usabilidade, disponibilidade, tolerância a falhas, segurança, modificabilidade e testabilidade.
  - Hard skills: Capacidade de projetar sistemas robustos "by design", mesmo lidando com atributos em trade-off.
  - Soft skills: Habilidade de envolver *stakeholders* técnicos e de negócio na importância de construir sistemas com robustez e controle de riscos, especialmente considerando escalabilidade digital e segurança cibernética.
- Visão de Engenharia: Aborda estratégias, táticas e abordagens para equilibrar decisões arquiteturais que agregam valor ao negócio.
  - Hard skills: Aporte de conhecimento para implementar táticas de detecção, recuperação e predição dos níveis de serviço estabelecidos nos requisitos não funcionais.
  - Soft skills: Motivação das equipes para adotar mecanismos sofisticados que atendam aos requisitos arquiteturais, liderando internamente.
- Visão de Tecnologia: Foca nas ferramentas, plataformas, componentes e serviços necessários para executar as funcionalidades planejadas.
  - Hard skills: Domínio e integração adequada de tecnologias para atender aos níveis de serviço refletidos nos requisitos não funcionais.
  - Soft skills: Capacidade de selecionar e justificar escolhas tecnológicas, comunicando efetivamente com diferentes stakeholders.

A evolução do aluno em aprendizado sobre sistemas digitais com arquitetura suficiente para atender às necessidades de negócio, tem como meta alcançar uma capacidade em: analisar, avaliar e criar soluções que suportam de maneira controlada as demandas e os riscos que podem afetar a robustez de processamento. O gráfico de radar apresentado na Figura 1 mostra as dimensões refletindo as visões arquiteturais citadas nos itens anteriores.

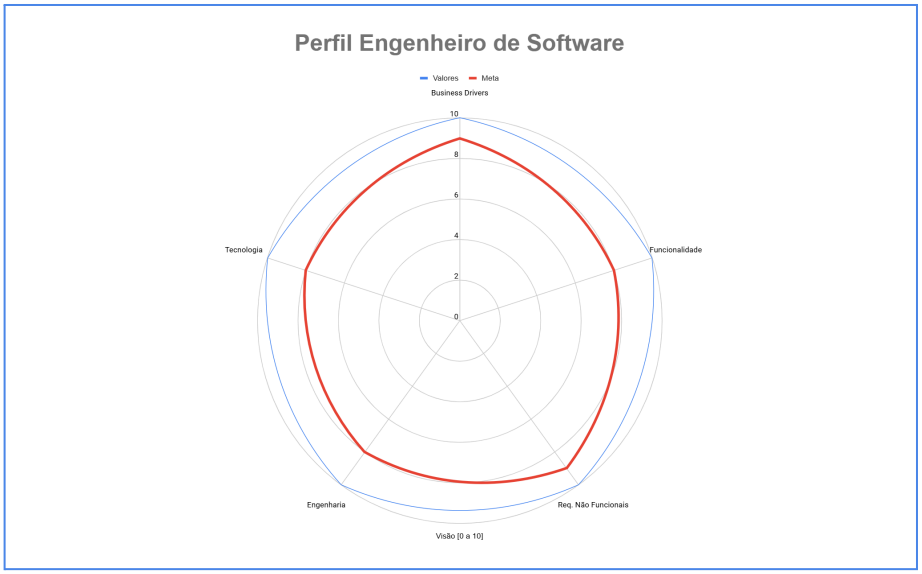
Os critérios atribuídos para cada uma das dimensões são comentados tomando as referências de Taxonomia de Bloom e Visões Arquiteturais, conforme ilustrado na Tabela 2.

Tabela 2 - Critérios de pontuação de conhecimento das visões arquiteturais

Visão de Arquitetura	Valores	Meta
Business Drivers	10	**9**
Funcionalidade	10	**8**
Req. Não Funcionais	10	**9**
Engenharia	10	**8**
Tecnologia	10	**8**

Fonte: Os autores.

Figura 1 – Gráfico de radar com visões arquiteturais e metas de formação em Engenharia de Software



Fonte: Os autores.

A meta, com a pontuação, é baseada em critérios, conforme indicado nos itens a seguir. A proficiência de aprendizado é destacada desde o conhecer até a fluência em analisar, avaliar e criar soluções de sistemas, expandindo os conceitos associados a cada uma das visões arquiteturais.

Os critérios estabelecidos a seguir ajudam a medir os níveis de aprendizado proporcionados pelos módulos, visualizar a evolução na formação do perfil do Engenheiro de Software e, com isso, medir e aprimorar os módulos baseados em tais referências.

- **Critérios na visão de Business Drivers:**

0~2: Entender e explicar um fluxo de processo de negócio, incluindo a parte a ser digitalizado por sistemas;
2~4: Detectar como o digital adiciona valor para o negócio;
4~6: Analisar volume, regras de negócio, usabilidade como elementos da solução digital;
6~8: Criar os requisitos funcionais e não funcionais combinando os itens de negócio, incluindo os riscos e não riscos e tradeoff, além da adição de valor digital;
8~10: O engenheiro sabe como transformar os itens de negócio como decisão arquitetural controlados digitalmente (cyber), ou seja com mecanismos para detectar, resistir, recuperar e rastrear os eventos que trazem a robustez do sistema, controlando os riscos de instabilidade.

- **Critérios na visão Funcionalidade:**

0~2: Entender uma tela e botão, conforme especificado e codificar as telas que implementa esta funcionalidade de acordo com a tecnologia disponibilizada;
2~4: Entende a modelagem: negócio, regra de negócio, fluxo de sucesso, fluxo de exceções;
4~6: Consegue analisar a qualidade com base em revisão estática e dinâmica - massa de testes e cobertura de testes;
6~8: Avalia a evolução e qualidade das funcionalidades aplicando conceitos de gestão de configuração para códigos e documentos e controle de qualidade;
8~10: A suficiência em conhecimento advém do controle de qualidade baseada em código, no qual a documentação é construída como ativo de software - códigos e massa de testes automatizados, com recursos de regressão para aferição da extensão e impactos de mudanças.

- **Critérios na Visão Requisitos Não Funcionais**

0~2: Entender as integrações internas e externas: códigos e componentes internos e terceirizados, controlando a evolução de versões de sistemas operacionais e produtos;
2~4: Controlar o desempenho e robustez dos requisitos funcionais por testes não funcionais e integrações de serviços internos e externos;
4~6: Avaliar as táticas arquiteturais e mecanismos não funcionais - medição de requisitos não funcionais;
6~8: Avaliação ATAM da arquitetura e revisão da arquitetura de sistemas (mapa de riscos e controle de qualidade);
8~10: Gestão da qualidade da arquitetura e tradeoffs, por automação com foco em negócios (Embarcar mecanismos de avaliação da arquitetura, baseada nos business drivers);

- **Critérios na Visão Engenharia:**

0~2: Testar, documentar e validar as integrações, com abrangências internas, externas;
2~4: Idem anterior para os requisitos não funcionais críticos;
4~6: Medir as táticas arquiteturais e mecanismos não funcionais - Estático e dinâmico;
6~8: Avaliação ATAM da arquitetura e revisão da arquitetura de sistemas (mapa de riscos e controle de qualidade) - focando os códigos (revisão e testes);
8~10: Gestão da qualidade da arquitetura e tradeoffs, por automação com foco em negócios, embarcando mecanismos de avaliação da arquitetura e gestão de configuração, com possibilidades de uso de IA e Machine Learning para suportar os processos de operação e de desenvolvimento (MLOPS e DEVSECOPS);

- **Critérios na Visão Tecnologia:**

0~2: Testes gerados por equipes internas e terceiras/fornecedores;
2~4: Revisão da qualidade de códigos internos e terceirizados (Gestão configuração);
4~6: Garantir a gestão de ciclo de vida dos componentes e demais dependências;
6~8: Testes e gestão de configuração das mudanças com esteira DEVSECOPS, aplicando o controle de qualidade no processo e no produto;
8~10: Lidar com o aging de tecnologias com integrações de inovações com legado, controlando a diversidade tecnológica de ferramentas, linguagens, plataformas e integrações de hardware software e serviços externos, com foco nos riscos de mudanças e no controle de configuração.

4 PROPOSTA DE EVOLUÇÃO SISTEMÁTICA DE UM MÓDULO PBL

Um módulo PBL, estabelecido e definido de acordo com as Diretrizes Curriculares Nacionais (DCN/MEC), é suportado por módulos de aprendizagem denominados LBL (Learning Backlog) que são constituídos por um pacote de objetos de aprendizagem, incluindo artefatos de projetos, instruções de aula em salas invertidas, auto-estudos com referências científicas ou com práticas industriais (na forma de estudos de casos, por exemplo), além de exercícios conceituais e práticos. O método foi desenvolvido para apoiar a evolução destes pacotes de aprendizagem, observando tanto seus aspectos internos quanto as relações entre diferentes LBLs. Para isso, consideramos as três questões fundamentais apresentadas na Introdução, juntamente com as orientações que estabelecemos para guiar a construção do método, as quais serão detalhadas a seguir.

Como realizar a evolução sistemática dos módulos de aprendizagem baseados em projetos (LBLs)?

Para garantir a revisão eficaz dos LBLs, buscamos estabelecer um processo de avaliação repetível e objetivo. Isso envolve:

- Implementar procedimentos padronizados para avaliar a situação atual e identificar pontos de melhoria;
- Utilizar critérios que minimizem interpretações subjetivas;
- Empregar referenciais científicos para embasar as decisões;
- Reduzir a dependência de ferramentas tecnológicas específicas, que podem se tornar obsoletas rapidamente devido à dinâmica da indústria.

Quais aspectos da aprendizagem devem ser utilizados como base para compor os requisitos de revisão?

A revisão dos LBLs é fundamentada em dois pilares principais, orientados pode indicadores que facilitem as análises e decisões:

- A Taxonomia de Bloom, que abrange desde o conhecimento básico até a capacidade de aplicação, avaliação e criação;
- As dimensões arquiteturais de um sistema resiliente e robusto, alinhadas com critérios de engenharia, muito mais que as práticas tecnológicas de ferramentas efêmeras nos ciclos de evolução industrial.

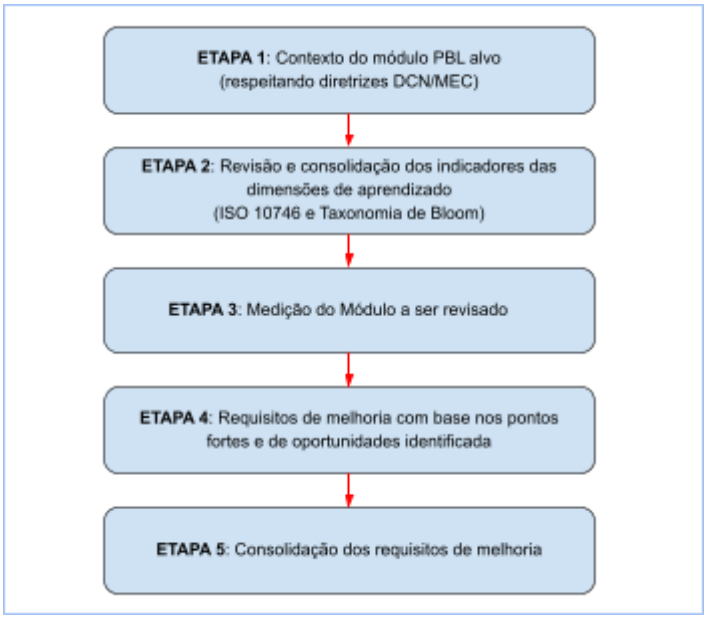
Como os pacotes de conceitos e práticas podem ser reconfigurados sem prejuízo das disciplinas essenciais definidas nas Diretrizes Curriculares Nacionais estabelecidas pelo MEC?

O diferencial da estrutura modular por LBL é a "componentização" do ensino, uma abordagem inovadora do Inteli. Esta abordagem flexível possibilita a reconfiguração dos pacotes de aprendizagem, otimizando o processo educacional sem comprometer os requisitos curriculares essenciais. Esta estrutura permite:

- Ajustar componentes com conteúdos de instrução, auto-estudos de conceitos e práticas;
- Buscar a melhoria contínua da aprendizagem projetada para cada módulo;
- Manter a conformidade com as disciplinas estabelecidas nas DCN/MEC, respeitando suas dependências.

Considerando isso, a metodologia proposta é constituída das seguintes etapas, conforme a Figura 2.

Figura 2 - Etapas para a revisão de um módulo PBL

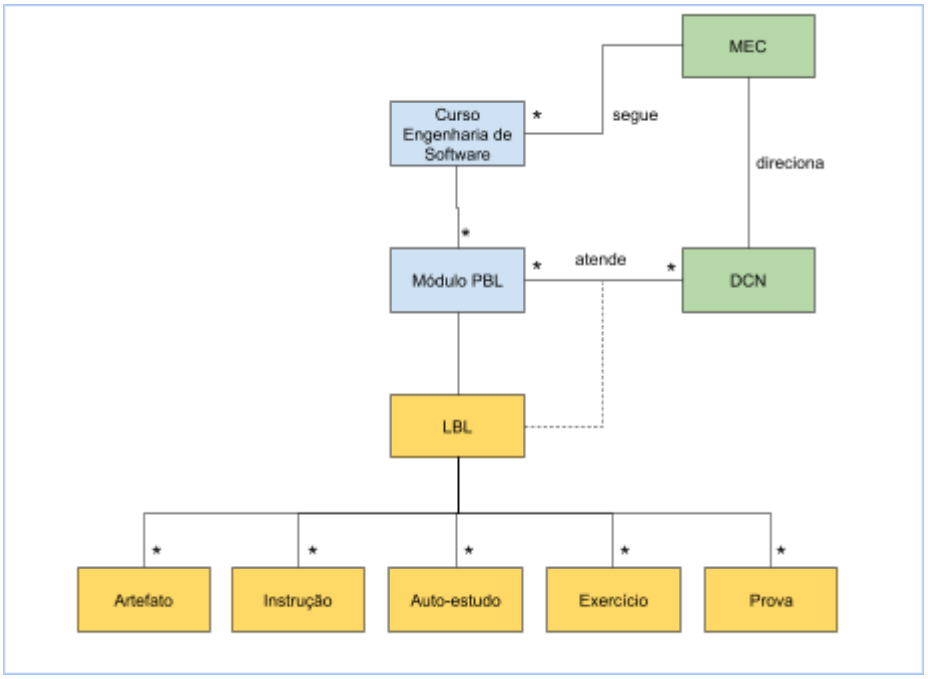


Fonte: os autores.

Na Etapa 1, os módulos PBL projetados formam um caminho ao longo do curso de Engenharia de Software para avançar na formação do perfil profissional do aluno. Esse contexto é importante para o alinhamento do contexto da revisão do LBL correspondente ao módulo alvo, sem colocar os aspectos das DCN/MEC. Os critérios e indicadores relacionados à arquitetura e taxonomia de Bloom são avaliadas para ver se há alguma necessidade de ajustes, com o devido racional das DCN na Etapa 2. Já na Etapa 3, revisadas as réguas de qualidade de aprendizagem, medir o módulo alvo da revisão de modo a mapear os pontos fortes e de oportunidades de melhoria, segundo percepções dos professores coordenador, instrutores dos eixos de UX, Matemática e Física, Negócios, Liderança e Computação, além das interações e avaliações dos alunos registrados na plataforma de ensino. Com o material resultante até aqui, passa-se para a Etapa 4, onde se consolidam os requisitos, destacando os pontos fortes e os pontos de oportunidades, foco da melhoria. Para partir para as ações de revisão, na Etapa 5, se consolidam os requisitos de melhoria, priorizadas e selecionadas para que os times de professores possam atuar a revisão do módulo, aprimorando as instruções, auto-estudos e exercícios de conceitos e da prática de projetos constantes no LBL.

Como conclusão, um paralelo pode ser visto pelo modelo UML sobre a estrutura de módulos PBL do curso de Engenharia de Software do Inteli e com isso, o mapa de impacto das revisões de maneira “componentizada”, explorando os aspectos de baixo acoplamento e alta coesão, quesitos importantes de um sistema que precisa ser modificado constantemente. Veja a Figura 3.

Figura 3 - Baixo acoplamento e alta coesão da LBL em relação aos módulos PBL e DCN/MEC



Fonte: os autores.

Na Figura 3, o diagrama mostra o curso alinhado com as diretrizes curriculares nacionais (DCN) através das disciplinas organizadas nos módulos, garantindo *compliance* com os requisitos do MEC. Como se fosse a componentização de software, o LBL é um componente com alta coesão com os objetos de aprendizagem relacionados aos artefatos de projeto, instruções na sala de aula invertida, auto-estudos que trazem os conceitos e

práticas, os exercícios de práticas e de avaliações, além das provas com questões que avaliam segundo as DCN.

5 RESULTADOS OBTIDOS

A aplicação do método proposto para evolução sistemática de módulos PBL resultou em avanços significativos na estruturação e revisão dos Learning Backlogs (LBLs) utilizados no curso de Engenharia de Software. A sistemática, composta por cinco etapas claramente definidas, demonstrou viabilidade prática e coerência conceitual, permitindo maior controle e clareza no processo de aprimoramento pedagógico.

Entre os principais resultados observados estão:

- Padronização do processo de revisão: A sistemática proposta permite que diferentes times docentes avaliem e aprimorem os LBLs de forma uniforme, reduzindo a subjetividade na identificação de lacunas de aprendizagem e promovendo alinhamento entre os módulos.
- Alinhamento entre competências e objetivos de aprendizagem: A combinação da taxonomia de Bloom com visões arquiteturais possibilitou a definição de critérios claros para avaliação do progresso dos alunos, considerando desde o domínio conceitual até a capacidade de criação e solução de problemas complexos.
- Componentização e modularidade pedagógica: A abordagem baseada em LBLs, com foco em baixo acoplamento e alta coesão, favoreceu a manutenção da aderência às Diretrizes Curriculares Nacionais (DCNs), ao mesmo tempo em que viabilizou adaptações e melhorias contínuas, sem comprometer a integridade do currículo.
- Clareza nos requisitos de melhoria: O método permitiu transformar a percepção dos envolvidos (docentes, coordenadores e alunos) em requisitos objetivos de evolução, com base em critérios técnicos e pedagógicos. Isso fortalece a tomada de decisão e facilita a priorização de ações de melhoria nos módulos.
- Base para automatização futura: A estrutura sistemática, baseada em indicadores observáveis e métricas cognitivas e arquiteturais, fornece um alicerce sólido para o desenvolvimento de ferramentas automatizadas de apoio à revisão curricular.

Embora este artigo não apresente um estudo de caso completo, os testes exploratórios com aplicação parcial da metodologia indicaram um ganho expressivo em organização, clareza de objetivos e qualidade pedagógica dos módulos analisados. A experiência inicial reforça o potencial do método como instrumento institucional de governança pedagógica para cursos baseados em PBL.

6 CONSIDERAÇÕES FINAIS

Neste estudo, foi criado um método sistematizado para melhorar os módulos de Aprendizagem Baseada em Projetos (PBL) em programas de Engenharia de Software. Ele combina a taxonomia de Bloom com perspectivas arquiteturais baseadas em padrões globais. Essa abordagem permite a avaliação objetiva dos PBLs, resultando em uma formação mais sólida e alinhada com as Diretrizes Curriculares Nacionais (DCN/MEC). Ao usar elementos arquiteturais como base para o desenvolvimento de habilidades técnicas e

socioemocionais, oferece-se uma visão estruturada do percurso educacional dos alunos. Os resultados indicam que essa metodologia promove consistência pedagógica e facilita a identificação de áreas para melhorias nos módulos. Uma contribuição significativa foi a criação de uma ferramenta prática e replicável para auxiliar na tomada de decisões sobre a atualização curricular, mantendo a flexibilidade e a qualidade do ensino baseado em projetos. Como próximas evoluções planeja-se automatizar partes destas revisões e o alinhamento com dados e percepções coletadas a partir das aplicações destas revisões junto aos estudantes e empresas parceiras, já em andamento.

AGRADECIMENTOS

Os autores agradecem à estrutura executiva e de coordenação do Curso de Engenharia de Software do Instituto INTELI pelo o apoio nos estudos, discussões e revisões dos conteúdos dos módulos PBL, sempre buscando o alinhamento junto ao MEC e observando os resultados das atividades de aprendizagem dos alunos e interações com empresas parceiras.

Os autores também agradecem o acolhimento e o apoio da equipe de professores de pós-graduação no ensino superior em PBL, pela busca incessante de desafios para a aprendizagem e aprimoramento das técnicas em salas de aula invertida.

REFERÊNCIAS

APEANTI, Wilson Osafo; ESSEL, Daniel Danso. Learning Computer Programming Using Project-Based Collaborative Learning: Students' Experiences, Challenges, and Outcomes. August. International Journal for Innovation Education and Research 9(8):191-207, 2021.

ARAKAKI, Reginaldo *et al*. Avaliação do oferecimento a distância de laboratório de eletrônica digital por meio de objetivos de aprendizagem e métricas do AVA. In: ANAIS do XLIX Congresso Brasileiro de Educação em Engenharia, 2021.

BARROWS, Howard. S. Problem-Based Learning in Medicine and Beyond: A Brief Overview. New Directions for Teaching and Learning, 3-12, 1996;

CUGNASCA, Paulo Sérgio *et al.* O Papel do Laboratório de Eletrônica Digital como Disciplina Integradora de Competências para o Desenvolvimento de Projetos de Engenharia. *In:* 52º Congresso Brasileiro de Educação em Engenharia, Vitória, 2024.

FELDER, R.; SOLOMAN, B. Index of Learning Styles (ILS). Disponível em: https://learningstyles.webtools.ncsu.edu/ Acesso em: 01/06/2025.

FIORAVANTE, M.L.; SENA, B.; PASCHOAL, L.N. Integrating project based learning and project management for software engineering teaching: An experience report. Proceedings of the 49th ACM Technical Symposium on Computer Science Education. Páginas 806-811. Editora ACM, 2018.

HABIMORAD, Maira; 2024. Acessar o site do Instituto Inteli. https://www.inteli.edu.br e solicitar **RELATÓRIO-ANUAL-2024-22082024-FINAL-DIGITAL.pdf**, Acessado na Internet em Janeiro 2025.

HAYASHI, V. et al. Laboratório Remoto para o Ensino de Engenharia**.** In: S WORKSHOPS DO CONGRESSO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO (WCBIE), 9. , 2020,

Online. Porto Alegre: Sociedade Brasileira de Computação, p. 187-194, 2020.

HAYASHI, V. T. et al. **Laboratório Virtual com Dados Reais.** In: SIMPÓSIO BRASILEIRO DE EDUCAÇÃO EM COMPUTAÇÃO (EDUCOMP), 1. , 2021, On-line. Anais [...]. Porto Alegre: Sociedade Brasileira de Computação, 2021. p. 296-304. DOI: https://doi.org/10.5753/educomp.2021.14497;

HAYASHI, Victor Takashi et al. Implementation of PjBL With Remote Lab Enhances the Professional Skills of Engineering Students. August 2023. IEEE Transactions on Education PP(99):1-10, 2023.

INTELI PPC. Acessar https://www.inteli.edu.br/engenharia-de-software. Projeto Pedagógico do Curso de Bacharelado em Engenharia de Software. 2024. Acesso em Janeiro de 2025.

ISO/IEC 12207:2008 Systems and software engineering – Software life cycle processes http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=43447;

ISO/IEC 25010:2011 - Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models http://www.iso.org/iso/catalogue_detail.htm?csnumber=35733;

ISO/IEC/IEEE 42010:2022. Systems and software engineering — Architecture description https://www.iso.org/standard/74393.html;

ISO/IEC 10746-2:2009 Information technology — Open distributed processing — Reference model: Foundations;

MARKHAM, T.; LARMER, J.; RAVITZ, J.; Aprendizagem Baseada em Projetos, Artmed Editora S/A, Porto Alegre, 2008.

PRESSMAN, Roger; MAXIM, Bruce; ARAKAKI, Reginaldo, (Revisão Técnica) Engenharia de Software: Uma Abordagem Profissional, Makron Book, January 2021.

ZHANG, M.; HU, W., (2024) Application of PBL combined with CBL teaching method in clinical teaching of vascular surgery. PLoS ONE 19(8), 2024.

SYSTEMATIC OPTIMIZING OF SOFTWARE ENGINEERING PBL (PROJECT BASED LEARNING) MODULE: A LEARNING GOALS-BASED METHOD USING ARCHITECTURE VIEWPOINTS

Abstract: This work details a methodology to review the learning modules backlog named LBL based on Project Based Learning (PBL), aligned with curricular rules and directions of MEC (Ministério de Educação) and PPC (Projeto Pedagógico do Curso) of Software Engineering Bachelor, considering industry professional profiles expected in terms of capacity, ability, and knowledge to build and maintain digital software platforms. This research drew on the experiences and disciplines of business, leadership, computation, mathematics, physics, and user experience experts. Scientific architecture references include ISO/IEC 25010, ISO/IEC 12207, and ISO/IEC 10746, which encompass five viewpoints of a digital system: Business Drivers, Functional and Non-Functional Requirements, Engineering Solutions, and Technology Tools.

Such dimensions of approach review are compatible with engineering aspects of modern digital platforms that lead with the human body, transportation devices, and connected cities all digitally integrated.

Keywords: Curriculum Alignment, Bloom's Taxonomy, Architecture Viewpoints, Modular Design, Learning Assessment, ISO/IEC 42010, Soft and Hard Skills, LBL (Learning Backlog), Project Based Learning.