UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DIRETORIA DE PESQUISA E PÓS-GRADUAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

NARCI NOGUEIRA DA SILVA

DESENVOLVIMENTO DE OBJETOS DE APRENDIZAGEM COM ELEMENTOS DE APRENDIZAGEM BASEADA EM PROBLEMAS NO CONTEXTO DE COMPUTAÇÃO PARA ENGENHARIAS

PROPOSTA DE DISSERTAÇÃO DE MESTRADO (PDM)
ORIENTADOR: DR. MARCO AURÉLIO GRACIOTTO SILVA

CORNÉLIO PROCÓPIO
FEVEREIRO 2016

NARCI NOGUEIRA DA SILVA

DESENVOLVIMENTO DE OBJETOS DE APRENDIZAGEM COM ELEMENTOS DE APRENDIZAGEM BASEADA EM PROBLEMAS NO CONTEXTO DE COMPUTAÇÃO PARA ENGENHARIAS

Proposta de Dissertação de Mestrado (PDM) apresentada como requisito parcial para obtenção do grau de mestre em Informática, do programa de Pós-Graduação em Informática da Universidade Tecnológica Federal do Paraná – Área de concentração: Computação Aplicada

Orientador: Dr. Marco Aurélio Graciotto Silva

CORNÉLIO PROCÓPIO

FEVEREIRO 2016

Resumo

Contexto: Estudantes normalmente tem dificuldades no entendimento dos conceitos básicos em disciplinas introdutórias à computação. Isso não acontece somente nos cursos da área de Informática, mas também nos cursos de outras áreas, como a Engenharia. Isso causa, além da dificuldade mencionada, dúvidas no porque aprender algoritmos e codificá-los em uma linguagem de programação, ocasionando desistência, desinteresse e dificuldade de alunos e professores em obter sucesso nas atividades em sala de aulas e laboratórios.

Diversas soluções são propostas na esfera educacional, tal como a aprendizagem baseada em problemas (PBL), com devido apoio de computação, para tratar desta questão. Em especial, no âmbito computacional, objetos de aprendizagem são empregados para fomentar tais iniciativas, promovendo a distribuição e reúso. Entretanto, embora existam iniciativas para o desenvolvimento sistemático de tais artefatos, a representação de aspectos interativos e, principalmente, de agregação de elementos educacionais modernos, constitui-se como um desafio para o desenvolvimento e reúso de objetos de aprendizagem.

Objetivos: Considerando este cenário, o problema a ser tratado é a necessidade de desenvolver objetos de aprendizagem para auxiliar e melhorar as formas de ensino existentes no contexto delimitado anteriormente. Desta forma, o objetivo deste mestrado é incorporar elementos relacionadas à PBL para o desenvolvimento de objetos de aprendizagem (OA) utilizando o método LODM (Learning Object Development Method).

Método: Executaremos as seguintes atividades iterativamente: estudar o LODM; estabelecer as características e requisitos de PBL; modelar elementos de PBL; criar o OA com PBL; aplicar o OA e avaliar se os resultados são adequados na visão dos professores e alunos.

Resultados esperados: Busca-se apresentar nesta Proposta de Dissertação de Mestrado novos mecanismos à LODM e o desenvolvimento de objetos de aprendizagem que possam não só auxiliar os alunos nos cursos de engenharia, mas também que eles possam entender o valor das disciplinas introdutórias na matriz curricular do seu curso e na profissão de engenheiro.

Abstract

Context: Students usually face difficulties understanding basic concepts of introductory computer courses. This is not restricted to Computing programs, but also affects Engineering graduation programs. Besides these hurdles, students do not perceive the importance on learning algorithms and programming, causing lower retention rate, lack of interest in classes, and low effectiveness of educational activities.
Several solutions have been defined in the educational setting, such as problem-based learning (PBL), with proper computing support, to tackle this issue. For instance, in the computational domain, learning objects can be employed to foster such initiatives, supporting distribution and reuse. However, despite the availability of approaches for systematically development of those artifacts, the representation of interactive concepts and novel educational elements is an important challenge for the development and reuse of learning objects.
Objective: Considering this scenario, the problem considered for this thesis is the need to develop learning objects that supports and improves current learning practices. Thus, our objective is to add support for elements related to PBL to the LODM, a learning object development method.
Method: In order to achieve our goal, we will apply an iterative approach, consisting of the following activities: study LODM, elicit requirements for PBL, define models for representing PBL elements, develop learning objects with PBL, use the learning objects, and evaluate their quality considering the viewpoint of students and professors.
Expected results: The current thesis proposal aims the development of new mechanisms for LODM regarding the development of learning objects that supports computer educations for Engineering programs, enabling a better understanding of the importance of Computing to future engineers.

Lista de figuras

Lista de tabelas

Sumário

Capítulo 1

Introdução

Computação é um instrumento amplamente utilizado em cursos de Engenharia, tanto diretamente, com ferramentas computacionais, como pelo desenvolvimento de raciocínio lógico e computacional necessários para resolver problemas ligados diretamente com a área de formação (MAHADEVAN-JANSEN *et al.*, 2003; SUN *et al.*, 2012; ZACHARY, 1996). Entretanto, o ensino e aprendizado de Computação, nesse contexto, não é trivial (NOOR *et al.*, 2014; AVOURIS *et al.*, 2010; EDGCOMB; VAHID, 2014; RECKINGER; RECKINGER, 2014).

De fato, disciplinas iniciais de Computação, mesmo em cursos específicos da área, apresentam números elevados de desistência e reprovação, ocorrendo a retenção de alunos nos cursos e ainda ocasionando o desinteresse desses (SARRIA; GERALDO, 2009; DYNE; BRAUN, 2014), dado que os alunos nem sempre entendem o porque de aprender lógica de programação, escrever algoritmos e codificá-los em uma linguagem de programação (LIANG *et al.*, 2013).

Neste contexto, nosso empenho será em buscar formas ou artefatos que mostre a estes alunos a real aplicabilidade da disciplina de programação em seus cursos, motivando-os a se engajarem e tornando-os mais receptivos e encorajando-os ao ensino, pesquisa e extensão, facilitando assim, sua formação plena e que sejam profissionais mais satisfeitos e realizados com suas

escolhas. Por exemplo, diversos estudos abordam a utilização do LEGO Mindstorms (LIANG *et al.*, 2013; DYNE; BRAUN, 2014; SARRIA; GERALDO, 2009), material disponibilizado na Web (NOOR *et al.*, 2014; EDGCOMB; VAHID, 2014), aprendizado baseado em problemas (PBL) (MAHADEVAN-JANSEN *et al.*, 2003; SUN *et al.*, 2012; ZACHARY, 1996) e utilização de linguagens de programação mais amigáveis, tais como Matlab (RECKINGER; RECKINGER, 2014; FAN; SCHWARTZ, 2003; SUN *et al.*, 2012), Maple (EDGCOMB; VAHID, 2014; SUN *et al.*, 2012; ZACHARY, 1996) e Python (AVOURIS *et al.*, 2010).

No âmbito computacional, objetos de aprendizagem são empregados como instrumento artefato de ensino. Existem iniciativas para o desenvolvimento sistemático de tais artefatos tais como IMA-CID (BARBOSA, 2004) e LODM (SILVA *et al.*, 2011). Todavia, a representação de aspectos interativos e, principalmente, de agregação de elementos educacionais modernos, tais como os citados anteriormente, constitui-se como um desafio para o desenvolvimento e reúso de objetos de aprendizagem. Considerando este cenário, o problema a ser tratado é a necessidade de desenvolver OA para auxiliar e melhorar as formas de ensino existentes no contexto delimitado anteriormente.

Portanto, nosso objetivo é propor um método de desenvolvimento de objetos de aprendizagem baseado em PBL, utilizando a abordagem LOD (SILVA, 2012) e a aplicação em disciplinas de Computação nos cursos de Engenharia. Especificamente, as metas são:

- Estabelecer as características de PBL, onde entenderemos melhor como aplicar a Aprendizagem Baseada em Problemas.
- Estabelecer os requisitos de PBL necessários para fazer parte do OA com enfoque no ensino de programação para alunos iniciantes nos cursos de engenharia.
- Modelar elementos de PBL para OA específico.
- Criar o OA com PBL, atendendo as características e requisitos estabelecidos, sempre levando em consideração os objetivos instrucionais da disciplina.
- Aplicar o OA em turmas de primeiros e segundos períodos dos cursos de engenharia do Campus Campo Mourão da UTFPR, visando coletar dados.
- Avaliar se os resultados são adequados na visão dos professores e alunos dos referidos cursos. Com isso vamos determinar o número de experimento vamos executar, quais cursos serão abrangidos.

Para justificar esta pesquisa, entendemos que é relevante na medida que se um OA for criado com todos as características e critérios, provavelmente trará benefícios para a aprendizagem e ainda será uma ferramenta de apoio para os professores. No âmbito computacional, a reutilização

destes OA em outros contextos e a redução do tempo de desenvolvimento são uma contribuição importante para a área.

A proposta esta dividida como segue. O Capítulo 2 trata do ensino de Computação para Engenharia, focando as características e dificuldades encontradas e mostrando um levantamento (*survey*) realizado no campus Campo Mourão da UTFPR e em uma revisão sistemática sobre ensino de Computação para Engenharia. No Capítulo 3 aborda a PBL e objetos de aprendizagem para ensino de Computação para Engenharia, focando os objetivos, características e experiências de PBL. Também Objetos de Aprendizagem com seus conceitos básicos, métodos e processos de desenvolvimento, LOD e LODM. O Capítulo 4 finaliza a proposta de dissertação de mestrado com o método utilizado para alcançar os objetivos propostos, algumas atividades e cronograma.

Capítulo 2

Ensino de Computação para Engenharia

Nos últimos anos o mundo mudou, trazendo novas formas de agir e a necessidade de adaptação à nova realidade. A forma de produção e consumo se torna global, a mudança traz incertezas e de certa forma também traz a insatisfação com o atual, fazendo com que cada vez mais se busque novas maneiras de viver, criar, agir, reagir, pensar e também estudar. Não mais só queremos um diploma, mas nos aperfeiçoar a cada dia, sendo os responsáveis pela nossa própria formação.

Estas mudanças na forma de ensinar nos cursos de graduação são lentas e novos modelos educacionais são difíceis de serem implementados. Novas metodologias estão sendo adaptadas da forma tradicional de ensinar e os professores tem que se adaptarem a elas. Freire (1996) diz que o educador precisa saber que "ensinar não é transferir conhecimento, mas criar as possibilidades para a sua produção ou a sua construção".

O ensino de computação para a Engenharia não uma tarefa fácil, pois geralmente é uma disciplina ofertada nos primeiros períodos e tem um conteúdo programático que envolve raciocínio lógico e computacional, que muitas vezes não são compreendidos ou aceitos pelos alunos causando reprovação ou desistência, conforme mostrado na Seção 2.1.

Diversas experiências vêm sendo realizadas e relatadas, mostrando novas formas para deixar as aulas mais interessantes, participativas, colaborativas, e trazendo aos alunos atividades relacionadas a sua vivência pessoal e ao seu curso de graduação. Para investigar tais experiências, dois estudos foram realizados. Primeiramente foi realizado um levantamento com alunos e professores dos cursos de Engenharia, cujo resultados são apresentados na Seção 2.2. Posteriormente, foi realizada uma revisão sistemática sobre ensino de computação para Engenharias, mostrada na Seção 2.3. Estes estudos mostram experiências no ensino de Fundamentos de Programação nos cursos de Engenharia, para alunos nos primeiros períodos. Os resultados destas investigações foram utilizados para subsidiar a escolha de elementos apropriados para tratar o problema deste trabalho de mestrado.

2.1. Computação para Engenharia

Computação é um instrumento amplamente utilizado em Engenharia, tanto diretamente, com ferramentas computacionais, como pelo desenvolvimento de raciocínio lógico e computacional necessários para resolver problemas complexos (WALKER, 2015). Desta forma, é requisito em cursos de graduação em Engenharia o oferecimento de disciplinas introdutórias à Programação (KICK; TREES, 2015).

Entretanto, o ensino e aprendizado de Computação nesse contexto não é trivial. De fato, disciplinas iniciais de Computação, mesmo em cursos específicos da área, apresentam números elevados de reprovação e afetam a retenção de alunos nos cursos (AURELIANO; TEDESCO, 2012). No âmbito das Engenharias, em suas diversas modalidades – Ambiental, Alimentos e Civil, para citar alguns exemplos –, os desafios são ainda maiores. Por exemplo, as disciplinas podem estar isoladas na matriz curricular dos referidos cursos, os alunos nem sempre entendem o porque de aprender lógica de programação, escrever algoritmos e codificá-los em uma linguagem de programação, ocasionando assim um possível desinteresse por parte destes alunos.

Este desinteresse, provavelmente provoca, em alguns cursos, especificamente em disciplinas introdutórias de programação, uma taxa de evasão é alta. Sendo assim, diversos estudos tentam encontrar soluções para diminuir essa evasão. Aureliano e Tedesco (2012) desenvolveram, entre 2002 e 2011, uma revisão literária com 79 artigos relacionados com ensino de informática, publicados em eventos nacionais relevantes, para propor estratégias para melhorar a qualidade de ensino e consequentemente a aprendizagem. Como resultado, observou-se um aumento no interesse da comunidade brasileira na área e foram apresentadas algumas ferramentas, linguagens de programação, metodologias e técnica de avaliação.

Walker (2015) argumenta que, no planejamento de qualquer curso, tem que se considerar os objetivos e o público-alvo e que o instrutor pode escolher uma grande variedade de temas e tópicos. A argumentação se concentra nas prioridades e envolve quatro pontos principais. No ponto 1, considera-se que em cursos não relacionados à computação não se deve utilizar tópicos de programação e sim temas como a computação na sociedade contemporânea. No ponto 2, comenta-se que a resolução de problemas algorítmicos com programação é apenas uma das muitas prioridades possíveis e que pode ser ensinada. No entanto, para ter um retorno razoável, os alunos envolvidos na programação teriam de aprender algumas sintaxe básica e aplicá-las a uma série de problemas. No ponto 3, argumenta-se sobre a relação entre prioridade e compromisso, apontando que os cursos devem suportar outras áreas, a matemática por exemplo. No ponto 4, defende-se que não é necessário ser um programador de computador, muito menos um cientista da computação, para escrever código: "os programas são simplesmente mais um meio através do qual podemos criar e expressar ideias de todo o espectro do pensamento humano". Como exemplos adicionais, os currículos de matemática podem usar MatLab ou outros pacotes de software em aulas introdutórias (e ou avançadas).

Ainda considerando a importância da Computação para a formação dos futuros profissionais, Kick e Trees (2015) apresentam o AP CSP (*Advanced Placement – Computer Science Principles*/Colocação Avançada – Princípios de Ciência da Computação). Nesta iniciativa, a computação é tratada como tema essencial, abordando tópicos atuais relacionados à área. O AP CSP integra seis práticas de pensamento computacional (Conexão à Computação, Criação de Artefatos Computacionais, Abstração, Análise de Problemas e Artefatos, Comunicação e Colaboração) com sete grandes ideias ou foco (Criatividade, Abstração, Dados e Informações, Algoritmos, Programação, Internet e Impacto Global). Desta forma, o curso é projetado para ajudar os alunos a compreender que, independentemente da área de estudo escolhida, a computação irá desempenhar um papel importante. Os conceitos ensinados no AP CSP vão ajudar a construir uma base de práticas computacionais que irão preparar alunos e professores para interagir com a tecnologia e, possivelmente, até mesmo contribuir para a criação e desenvolvimento de inovações computacionais que tenham efeito positivo na sociedade, economia ou cultura.

2.2. Levantamento para caracterizar o ensino de Computação para Engenharia

A Computação é amplamente utilizada como disciplina introdutória em cursos de Engenharias, auxiliando no desenvolvimento raciocínio lógico e computacional. Entretanto, seu ensino não é

uma tarefa trivial, ocasionando números elevados de reprovação. Apresentamos neste Survey uma caracterização preliminar sobre disciplinas introdutórias à Computação nos cursos de Engenharia, observando a necessidade da disciplina, sua relevância nas atividades de ensino, pesquisa e extensão, e, finalmente, os fatores que afetam, positiva ou negativamente, o ensino de Computação no cursos de Engenharia.

Considerando este cenário, relatamos um levantamento sobre as disciplinas de programação nos semestres iniciais dos cursos de Engenharia Ambiental, Engenharia de Alimentos e Engenharia Civil de nossa instituição. Como isto, objetivamos identificar características dos oferecimentos dessas disciplinas, em especial a importância de Computação para Engenharia e quais os fatores que afetam, positiva ou negativamente, tal disciplina nas perspectivas dos alunos, dos professores e dos coordenadores, esperamos a delimitação dos problemas enfrentados, fomentando a escolha e definição de mecanismos para auxiliar no sucesso de tais disciplinas.

2.2.1. Métodos e Procedimentos

Neste levantamento foram disponibilizados, via Web, quatro questionários para alunos, coordenadores, professores do departamento de computação e professores de outros departamentos.

Os questionários foram enviados para: 120 alunos dos últimos dois períodos dos cursos de Engenharia de Alimentos, Ambiental e Civil, pois já foram aprovados nas disciplinas pesquisadas e ainda tem vivência no ensino, pesquisa, extensão e estágio; 7 coordenadores dos referidos cursos de Engenharias, pois tem a experiência além de sala de aula e são membros dos Colegiados de Curso e NDE (Núcleo Docente Estruturante), fornecendo uma visão ampla dos cursos; 25 professores do Departamento de Computação, pois tem o contato direto em sala de aula com os alunos e podem contribuir relatando essa vivência; 8 professores de outros Departamentos que ministram disciplinas que possuem disciplinas de Computação como pré-requisitos, tal como Cálculo Numérico, Geoprocessamento e Teoria de Estruturas.

O questionário foi ensaiado com uma aluna e uma professora de outra instituição, por não fazer parte da amostra apresentada. Com base neste, foram realizadas pequenas correções para que o entendimento das questões. Os modelos dos questionários enviados para todos os atores estão no final desta proposta, nos apêndices.

Para a execução do estudo, enviamos um convite por e-mail para todas as entidades relacionadas para participar da pesquisa respondendo os questionários. As respostas do coordenadores, professores e alunos não foram identificadas pelo questionário, preservando-se assim a anonimidade.

Os questionários ficaram disponíveis por duas semanas. Durante este período, foi monitorada a quantidade de questionários preenchidos e enviados lembretes sobre os questionários a serem respondidos. Após o prazo final determinado, os questionários foram fechados, permitindo a análise dos dados e das respostas, conforme apresentados a seguir.

2.2.2. Resultados

Os questionários foram respondidos por 14 alunos (aproximadamente 11,6%), 5 coordenadores (71,4%), 12 professores de Computação (48%) e 5 professores de outras disciplinas (62,5%). Quanto aos alunos, 6 cursam Engenharia Ambiental, 5 de Engenharia de Alimentos e 3 de Engenharia Civil.

2.2.2.1. Importância da disciplina quanto ao ensino

Procurando responder se a disciplina introdutória à Computação é necessária para o curso de Engenharia Ambiental, Engenharia de Alimentos e Engenharia Civil, foi indagada a relevância da disciplina para o curso nas atividades de ensino, pesquisa e extensão e ainda quais são os fatores que afetam, positiva ou negativamente, o ensino de Computação em relação ao curso. Analisando as respostas aos questionários, identificamos o percentual de importância na visão dos alunos (questão 6 - apêndice I), coordenadores (questão 2 - apêndice II) e professores (questão 7 - apêndice II). Desta forma, agregamos na Tabela 2.1 o resultado desta questão.

Tabela 2.1: Importância da disciplina de Computação para os cursos de Engenharia segundo alunos, coordenadores e professores de Computação.

Importância	Alunos	Coordenadores	Professores (Computação)
Extremamente importante (essencial)	7,14%	25,00%	33,33%
Importante	57,15%	75,00%	66,67%
Nada importante	35,71%	0,00%	0,00%

Analisando o questionário dos alunos, os percentuais e as respostas das questões abertas e comentários, percebemos que os alunos consideram que a disciplina *"ajuda no desenvolvimento do raciocínio lógico, visão crítica e ajuda no entendimento de termos da área profissional"*, mas necessita de *"melhor esclarecimento onde aplicar no curso e no futuro profissional"*. De fato, adquirir "raciocínio lógico" é um dos objetivos das disciplinas de Computação constantes no projeto pedagógico dos curso de Engenharia investigados.

Entretanto, aproximadamente 35% dos alunos não consideraram a disciplina importante para seu curso. Isso é preocupante, pois evidencia uma deficiência no aproveitamento da disciplinas e

que as competências que deveriam ser desenvolvidas não são vislumbradas ou aproveitadas por estes alunos nas disciplinas subsequentes do curso. Em contraposição aos aspectos de ensino, algumas respostas de alunos afirmam que a disciplina proporciona benefícios em atividades de estágio e pesquisa, como segue : *"O projeto do qual participo tem como objeto de estudo a aplicação de métodos numéricos em engenharia. A importância da programação neste projeto se justifica pela necessidade da implementação de algoritmos para a realização de uma grande quantidade de cálculos, gerada pela aplicação de tais métodos da forma mais precisa possível"*. Por fim, percebemos que alguns alunos reprovam nas disciplinas, ficam com a dependência, precisam da disciplina para passar pelo pré-requisito e avançar no curso e, mesmo assim, consideram a disciplina de programação nada importante para o seu currículo.

Os coordenadores concordaram com a importância das disciplinas de Computação para seus cursos, conforme percentual de 25% demonstrado na Tabela 2.1, e ainda comentaram que *"Esta disciplina se torna a base para demais disciplinas que compõem a matriz curricular dos cursos de engenharia"*. Destacaram também a importância do raciocínio lógico e de programação: *"Essas habilidades são necessárias e serão associadas a outras disciplinas específicas do curso (Modelagem, Simulação e Controle de Processos, por exemplo)"* e ainda que *"há uma tendência do ensino por meio de softwares ou até mesmo no desenvolvimento deles por parte de alunos e professores"*.

Os professores de Computação reforçaram a importância das referidas disciplinas, mas apontaram dificuldades dos alunos em relação a disciplina: *"Em geral percebo que existe um 'bloqueio' pelos alunos em aprender a disciplina. Aparentemente existe a pergunta 'porque aprender algoritmos se eu faço engenharia?'*.

Em suma, quanto a importância da Computação para o curso, o projeto pedagógico e a coordenação atestaram a importância de computação concordando com o projeto pedagógico, mas os alunos não observaram a realização desta importância nas demais disciplinas da grade curricular.

2.2.2.2. Importância da disciplina para pesquisa e extensão

Além de atividades de ensino, durante a graduação os alunos podem realizar atividades de pesquisa e extensão, além de serem obrigados a realizar estágio. Dos alunos que participaram do estudo, 7 participaram de projetos de pesquisa, 5 de projeto de extensão e 6 alunos concluíram o estágio (alguns alunos realizaram atividades de pesquisa, extensão e estágio concomitante). Além disso, 3 alunos não fizeram pesquisa, extensão ou estágio ainda durante o curso.

Para melhor ilustrar a visão desses alunos em relação a importância das disciplinas introdutórias à Computação para seu curso considerando pesquisa, extensão ou estágio(questão 10 - apêndice I), utilizamos a Tabela 2.2. Reforçando resultados da seção anterior, percebemos que a maioria (57,14%) não considera Computação importante. Em contraste a este dado, uma parcela significativa dos alunos consideraram Computação relevante para tais fins, inclusive com mais importância do que aquela observada na Tabela 2.1, quanto ao curso como um todo.

Tabela 2.2: Importância da disciplina de Computação, segundo os alunos, em relação às atividades de pesquisa, extensão e estágio.

Importância	Contagem	Percentagem
Extremamente importante (essencial)	3	21,43%
Importante	3	21,43%
Nada importante	8	57,14%

2.2.2.3. Importância da Computação para Engenharia

Os professores de outros departamentos, que ministraram disciplinas que tem como pré-requisito as disciplinas de Computação, tais como Cálculo Numérico, Geoprocessamento e Teorias de Estrutura, corroboram com os comentários referentes às Questões 3 e 4 do Apêndice III e acrescentam especificidades de suas áreas, apontando a necessidade de banco de Dados e outros relacionados a Informática Básica. Também comentam as dificuldades dos alunos em algoritmos das técnicas matriciais: *"A dificuldade consiste no aluno desenvolver e implementar algoritmos das técnicas matriciais de análise de estruturas. Assim, durante o curso é dado ênfase ao fato de que na prática não há como não realizar análises mais complexas sem a utilização de ferramentas computacionais"*. Alegaram ainda que poderiam ser utilizadas outras linguagens de programação, sejam linguagens de propósito geral como Python, C ou C++, aquelas de cunho específico de engenharia: *"Com relação à programação eles tem noções de programação que podem ser extrapoladas para a linguagem que usamos (LEGAL - Linguagem Espacial para Geoprocessamento Algébrico)"*.

Em relação à Informática Básica, eles observaram que os alunos tem formação heterogênea e que seria necessário um nivelamento ou curso de extensão: *"Nossos alunos tem formação básica bem heterogênea (o mesmo pode ser dito do nível social) [...]. O ideal seria se existisse algo extra para nivelamento desses alunos (como um curso de informática básica fora da grade)"*.

Além dos pré-requisitos explícitos, os professores de outros departamentos citaram as disciplinas de Cartografia e Sensoriamento Remoto como disciplinas que dependem de conhecimento e habilidade de Computação/Programação. Em relação à projetos de Pesquisa e Extensão

(Questão 5 - Apêndice IV), eles relatam que as dificuldades foram as mesmas encontradas quanto no ensino (Questão 3 - Apêndice IV), acrescida a falta de conhecimento em Banco de Dados.

2.2.2.4. Importância de conceitos de Computação

Na questão 8 - apêndice I, os alunos deveriam ordenar alguns conceitos de Computação de acordo com a importância para a formação de Engenheiro. Os conceitos, conforme definido pelo AP CSP, eram: Criatividade, Abstração, Dados e Informações, Algoritmo, Programação, Internet e o Impacto Global. Os resultados são mostrados na Tabela 2.3. A classificação 1 é a mais importante e mostra que os alunos consideram Internet e Impacto Global como os conceitos mais importantes para a formação de engenheiros. Abstração e Algoritmos foram considerados os conceitos menos importantes. Isto mostra uma desarmonia com o ementário das disciplinas de Computação, que dentre alguns itens, destacamos: *"estudo das formas de representação do pensamento lógico através de técnicas de desenvolvimento de algoritmos; e métodos, técnicas e processos de desenvolvimento de software"*. Em contrapartida, elementos que não constam no ementário, como "Internet" e "Impacto global", foram definidos como importantes.

Tabela 2.3: Conceitos importantes para a formação de Engenheiro segundo os alunos.

Conceito	1ª Classificação		2ª Classificação		3ª Classificação	
	Qtde	%	Qtde	%	Qtde	%
Internet	4	28,57%	2	14,29%	1	7,14%
Impacto global	4	28,57%	2	15,29%	1	7,14%
Dados e informações	3	21,43%	5	35,71%	1	7,14
Criatividade	2	14,29%	2	14,29%	3	21,43%
Programação	1	7,14%	1	7,14%	4	28,57%
Abstração	0	0,00%	0	0,00%	4	28,57%
Algoritmos	0	0,00%	2	14,29%	0	0,00%

2.2.2.5. Dificuldades de ensino

A maioria dos professores da área de Computação ministraram mais de três vezes as disciplinas de Computação para os cursos de Engenharia (questão 1 - apêndice III). Ao serem solicitados para ordenar as principais dificuldades encontradas nestas disciplinas (questão 3 - apêndice III), podemos observar nos resultados (Tabela 2.4) que a principal dificuldade refere-se ao "comportamento e interesse dos alunos". Na segunda classificação verificamos que "Abordagem e Turmas misturadas" tem 27,27% e na terceira classificação "Alunos em dependência" tem 30% seguida de Abordagem com 20%.

Tabela 2.4: Principais dificuldades encontradas pelos professores de Computação.

Dificuldade	1ª Classificação		2ª Classificação		3ª Classificação	
	Qtde	%	Qtde	%	Qtde	%
Comportamento e interesse dos alunos	10	83,33%	1	9,09%	1	10%
Alunos em dependência	1	8,33%	2	18,18%	3	30%
Abordagem	0	0,00%	3	27,27%	2	20%
Turmas misturadas	0	0,00%	3	27,27%	1	10%
Programa	0	0,00%	2	18,18%	1	10%
Linguagem de programação utilizada	1	8,33%	0	0,00%	1	10%
Mecanismos de avaliação	0	0,00%	0	0,00%	1	10%

2.2.2.6. Instrumentos utilizados para o ensino

Na Tabela 2.5, os professores de Computação ordenaram de acordo com a importância os instrumentos e técnicas utilizados em sala de aula(questão 5 - apêndice III). Com 25%, quadro branco e projetor multimídia foram os instrumentos mais utilizados. Simulação e Visualização de Algoritmo e Aprendizagem Baseada em Problemas, com 25 e 16,67% respectivamente, são as principais técnicas utilizadas. Em contrapartida, TDD (Test-Driven Development), Competições de Programação e Programação em Pares não são mecanismos comumente utilizados. Na segunda classificação verificamos que o Quadro branco aumentou para 33,33% e Projetor multimídia diminuiu para 16,67%, ainda TDD (Test-Driven Development), Competições de Programação e Programação em Pares estão em 0,00%. Na terceira classificação, Exemplos tem 41,67%, seguido por Quadro branco e Projetor multimídia com 16,67%, ainda empataram com 8,33% TDD (Test-Driven Development), Aprendizagem baseada em problemas e Programação em Pares. Desta forma, observamos que são frequentemente utilizados recursos tradicionais aliados com simulação e visualização de algoritmos, mas pouca ênfase em técnicas mais avançadas, tais como aprendizagem baseada em problemas, TDD, programação em pares e competições.

Tabela 2.5: Instrumentos e técnicas utilizados nas aulas de Computação.

Instrumento ou técnica	1ª Classificação		2ª Classificação		3ª Classificação	
	Qtde	%	Qtde	%	Qtde	%
Quadro branco	3	25,00%	4	33,33%	2	16,67%
Projetor	3	25,00%	2	16,67%	2	16,67%
Simulação e visualização de algoritmos	3	25,00%	2	16,67%	0	0,00%
Exemplos	1	8,33%	3	25,00%	5	41,67%
TDD (Desenvolvimento baseado em testes)	0	0,00%	0	0,00%	1	8,33%
Competições de programação	0	0,00%	0	0,00%	0	0,00%
Aprendizagem baseada em problemas	2	16,67%	1	8,33%	1	8,33%
Programação em pares	0	0,00%	0	0,00%	1	8,33%

2.2.3. Ameaças à validade

A quantidade baixa de respostas, principalmente no questionário do Alunos, é um fator preocupante em relação a validade. Fizemos uma rodada experimental, enviando os questionários para avaliação de alguns membros dos grupos pesquisados.

Estes resultados preliminares são aplicáveis no contexto da UTFPR de Campo Mourão, não podendo ser generalizado. Contudo o mesmo cenário será considerado no restante do trabalho e provavelmente, guarda semelhança com outros cursos oferecidos na UTFPR e ainda em outras instituições de ensino superior.

2.3. Revisão sistemática sobre ensino de Computação para Engenharia

O principal objetivo desta Revisão Sistemática foi fornecer uma visão geral de trabalhos publicados sobre o ensino de programação para alunos iniciantes em cursos de Engenharia, apresentando cenários, métodos, modelos, ferramentas de ensino para melhoria no ensino aprendizagem. Para isso um mapeamento foi realizado com o retorno de 97 artigos, onde todos foram analisados e ao final 13 artigos, que atenderam os critérios de Intervenção e População, foram selecionados. A pesquisa levou em consideração além da intervenção e população, os resultados ou seja, se foi aplicado algum experimento ou técnica e se resultados foram apresentados.

A revisão sistemática da literatura é uma forma de pesquisar e avaliar artigos publicados e que tenham importância para um trabalho ou área de interesse. Estas são desenvolvidas a fim de resumir evidências existentes, identificar eventuais lacunas em pesquisa ou ainda fornecer estrutura para apoiar novas investigações (KITCHENHAM; CHARTERS, 2007), reduzindo tendências na seleção de pesquisas (KITCHENHAM *et al.*, 2004; KITCHENHAM; CHARTERS, 2007; BIOLCHINI *et al.*, 2005).

Também necessita de uma questão primária, fazendo parte do protocolo da revisão sistemática. Para atender os objetivos deste trabalho a questão primária e: Como poderia ser alterado o LODM para o desenvolvimento de OA baseado em PBL a ser aplicado em turmas inicias dos cursos de Engenharia?

Considerando a questão primária, caracterizamos a população, intervenção e resultados esperados, como segue:

- População: alunos de Engenharia.
- Intervenção: disciplinas introdutórias de Computação.

- Resultados: criação de objeto de aprendizagem.

A expressão de busca foi executada na base de dados SCOPUS, pois indexa todas as revistas e algumas das principais conferencia de Educação e Engenharias da ACM e IEEE. Na expressão de busca foi utilizado operador booleano “AND” para selecionar os termos para a população, o “OR” para os termos para a intervenção e o “NOT” para excluir o termos “software engineering”, como segue na Figura 2.1.

```
((engineering) AND ("introduction to programming" OR "introduction to
computing" OR CS0) AND (not "software engineering"))
```

Figura 2.1: Expressão de busca na base de dados SCOPUS

A execução da expressão de busca, considerando o período de publicação de 1996 e 2014, encontrou 97 artigos foram encontrados. Após a leitura de títulos e resumos, 55 artigos foram excluídos por não fazer da População e/ou da Intervenção, restando 42 artigos pré-selecionados para leitura de todo o texto. Após leitura na íntegra destes 42 artigos, 29 artigos foram excluídos por constatar não fazer parte da População e Intervenção, por não mostrar qual foi a Intervenção ou quais ferramentas ou métodos foram aplicadas na Intervenção, artigos curtos, por não ser de Engenharia ou eram exclusivamente de Engenharia da Computação e ainda por não mostrar análise dos resultados. Na Tabela 2.6 é apresentada a relação dos artigos selecionados (ordem alfabética).

Na subseção a seguir, apresentamos um resumo dos 13 artigos selecionados, destacando a intervenção, o método de pesquisa e avaliação, e os resultados obtidos. Após esta apresentação, sumarizamos os resultados encontrados.

2.3.1. Artigos selecionados

ARTIGO 01 O artigo relata a experiência de integrar o LEGO Mindstorms com programação. A população do curso era aproximadamente de 500 alunos. O curso teve três características. A primeira, estes alunos não tinham conhecimento em programação; a segunda, eles eram de diversas áreas (Engenharia, Desenho Industrial, Computação) e a terceira característica, os alunos eram da china e estudavam em uma universidade Britânica (as aulas eram ministradas em Inglês). Quanto à intervenção, a linguagem de programação utilizada foi Java e a ideia era usar LEGO para ajudar os alunos na compreensão do paradigma de orientação a objetos. O LEGO foi visto como uma ferramenta pedagógica, onde foram usados todos os componentes

Tabela 2.6: Relação dos 13 artigos em ordem alfabética

Id	Título	Autores
01	A first introduction to programming for first-year students at a Chinese university using LEGO MindStorms	Liang *et al.* (2013)
02	A paradigm shift in the approach to freshman engineering education	Mahadevan-Jansen *et al.* (2003)
03	An interactive programming course model for mechanical engineering students	Reckinger e Reckinger (2014)
04	Application of the Pedagogical and Andragogical Model in Web-Based Learning Instruction Among Non-major Computer Science Students' Learning Programming	Noor *et al.* (2014)
05	Effectiveness of a Computational Thinking (CS0) Course on Student Analytical Skills	Dyne e Braun (2014)
06	Effectiveness of online textbooks vs. Interactive web-native content	Edgcomb e Vahid (2014)
07	First programming course in engineering: Balancing tradition and application	Fan e Schwartz (2003)
08	Hybrid course design: Leading a new direction in learning programming languages	Sun *et al.* (2012)
09	Introduction to Computing for Engineers: New Approaches to Content and Pedagogy	Zachary (1996)
10	Introduction to programming for engineers following the parachute paradigm	Sarria e Geraldo (2009)
11	MATLAB meets LEGO Mindstorms – A freshman introduction course into practical engineering	Behrens *et al.* (2010)
12	Reinventing CS50	Malan (2010)
13	Teaching introduction to computing through a project-based collaborative learning approach	Avouris *et al.* (2010)

LEGO (motor, sensores, etc), no ambiente de programação NXT-G. Como metodologia, o curso foi ministrado em sete semanas na segunda metade do semestre por dois semestres. Nas quatro primeiras semanas foram realizadas aulas com conceitos de programação (análise de programação e algoritmo, sequência de comandos, agrupamento de conceitos e reutilização, uso de condicionais, localização de erros). Nas três últimas semanas, os alunos foram divididos em 119 grupos em laboratório onde, utilizando LEGO, tinham que realizar duas atividades práticas. As equipes desenvolveram os códigos em NXT-G, embarcavam no LEGO e monitoravam o experimento com a supervisão de um dos autores do artigo.

Foram realizados questionários, entrevistas e observações dos grupos em laboratório executando as atividades com o LEGO para avaliar e analisar a dinâmica dos grupos e o trabalho em equipe. Após a coleta de 364 respostas dos questionários os resultados para as questões foram analisadas como segue:

1. **Eficiência global do Lego Mindstorms**.

Todas as equipes concluíram as duas atividades e a média das notas foi de 75% maior do que quando não usou LEGO, portanto a integração fez aumentar o conhecimento dos alunos. Comparando a pergunta *"I had some knowledge"* para o questionário antes do uso do LEGO, o percentual foi de 39,5 e depois do uso do LEGO foi de 62,3.

Nas questões a seguir foi utilizada a Escala de Likert, em que 1 representa "discordo completamente" e 7 "completamente de acordo", para recolher opiniões subjetivas dos alunos sobre os conjuntos LEGO.

2. **Lego Mindstorms proporcionou a uma experiência de aprendizagem positiva de conceitos de programação?**.
   Analisando as questões focando a aprendizagem, as respostas foi de 5,6.
3. **O Lego Mindstorms ajudou a adquirir conceitos específicos e fundamentais de programação?**.
   Com foco na compreensão dos conceitos básicos de programação e analisando se foi possível entender onde colocar os comandos certos no código, se ajudou a entender os comandos condicionais e se foi possível encontrar erros, os alunos constataram que foi possível compreender os comandos básicos da linguagem de programação, com a média de 5,57.
4. **Será que os alunos apreciam trabalhar em grupo?**.
   Como os alunos estavam no primeiro ano na universidade, a pesquisa queria investigar se foi agradável aos alunos trabalhar no seu grupo, se as atividades estavam bem organizadas e se todos se envolveram nas atividades. A média de 80% aproximadamente, com 5,57(pontos na Escala de Likert) endossando a respostas. Somente no item se todos os alunos estavam envolvidos plenamente com as atividades do grupo, com 75,7% ou 5,3 (pontos na Escala de Likert), que evidenciamos um valor menor.

O feedback dos alunos, sobre suas experiências, foi com o aumento nas notas médias em 75%. A maioria aprendeu os conceitos básicos, constataram que a aprendizagem foi positiva, que conseguiram aplicar os conceitos fundamentais da linguagem (comandos, expressões condicional e encontrar erro nos códigos), trabalharam bem em grupo e ainda ficaram por vontade própria mais tempo em laboratório por achar mais interessante aprender programação com a integração com o LEGO Mindstorms. Portando, os autores concluem que houve maior interesse, mais engajamento, menos medo de programação com a integração e, para o futuro, pretendem aumentar a dificuldade das missões tendo em vista que algumas equipes tiveram dificuldades com as engrenagens do LEGO e ainda aumentar o tempo, aumentar a carga horária.

ARTIGO 02 O artigo relata a experiência de um novo curso introdutório que foca a aprendizagem em desafios e abordagem de resolução de problemas de engenharia com e sem

o auxílio de ferramentas computacionais. Os alunos são apresentados a diferentes tipos de problemas de engenharia, específicos de cada disciplina, em cada um dos módulos. O rendimento do aluno é avaliado com base no processo de resolução de problemas, em vez de avaliação com resultado final.

Várias atividades em grupo foram desenvolvidas principalmente para reforçar a dinâmica do trabalho em pequenos grupos técnicos. Os autores listaram as etapas fundamentais do processo de resolução de problema, adotada no estudo:

1. Definir o problema.
2. Reunir informações pertinentes.
3. Gerar múltiplas soluções.
4. Analisar e escolher uma solução.
5. Testar e implementar a solução.

Os resultados preliminares, após pesquisa em sala de aula com 262 alunos selecionados dos 469 inscritos, indicam claramente uma maior consciência dos problemas de engenharia. Em particular, os aspectos de concepção do currículo de engenharia foi bem abordada como indicado por respostas dos alunos. Assim, acredita-se que a mudança de paradigma pode preparar melhor os estudantes para os anos seguintes como um estudante de engenharia e informar melhor os alunos sobre engenharia. Outras análises irão fornecer esclarecimentos adicionais sobre o processo, permitindo que o novo curso seja aperfeiçoado e utilizado no futuro.

ARTIGO 03 O artigo relata a experiência de um curso para alunos de Engenharia Mecânica nos EUA que trata a introdução à programação durante o primeiro ou segundo ano do curso. O objetivo do curso é proporcionar aos alunos o aprendizado de técnicas de programação básicas. Além disso, o curso visa desenvolver uma variedade de habilidades que transcendem todas as disciplinas científicas, incluindo a resolução de problemas, raciocínio lógico, depuração e treinamento de software.

O curso de programação foi desenvolvido para resolver os problemas existentes no modelo de curso original, que incluem:

- curso a ser oferecido fora do departamento de engenharia;
- variação no ritmo em que os estudantes compreendiam o material, e
- frustração dos novos programadores, especialmente com a depuração.

O novo curso enfoca aplicações específicas de engenharia, é ensinado por um professor da Engenharia Mecânica, e usa uma linguagem de programação mais prática, MATLAB. Assim, o conceito essencial da programação são introduzidos dentro de uma estrutura focada que cultiva o

desenvolvimento de ferramentas analíticas comumente usados em disciplinas de engenharia, tais como estatísticas, análise de dados, a diferenciação numérica e integração e análise de Fourier.

Em segundo lugar, o Process-Oriented Guided Inquiry Learning (POGIL), método orientado para o processo, é usado para que os alunos sejam auto-guiado através de parte da instrução.

Por último, o tempo de aula é organizado de tal forma que o instrutor gasta mais de metade do tempo de trabalho diretamente com indivíduos e pequenos grupos. Isto dá aos alunos uma oportunidade de ter explicações individualmente para o seu nível de compreensão, bem como tempo de sobra para assistência pelos pares e instrutor com a depuração.

O curso inicialmente ocorreu no início de 2013, durante 15 semanas e tinha 37 alunos divididos em duas seções diferentes. Não houve assistentes de ensino. O feedback dos estudantes indicaram que eles se beneficiaram muito com o projeto do curso. Melhorias para a segunda iteração do novo modelo de curso, que ocorreu no início de 2014, incluiu o aumento da duração do curso de 2,5 horas por semana para 4 horas, utilizando aula teórica mais tradicional, incorporando discussões em classe, a adição de conteúdo de vídeo suplementar criado pelos estudantes, e integrando um tema humanitário global para todas as atribuições em um esforço para apoiar as metas de artes liberais da universidade.

ARTIGO 04 O artigo analisa a aprendizagem dos alunos do ponto de vista pedagógico e andragógico. Foi desenvolvido um protótipo de ambiente de aprendizagem on-line aplicados a 32 alunos da faculdade de educação, na Malásia, para ensinar introdução a programação.

O modelo tradicional de ensino pedagógico foca na transmissão do conteúdo pelo professor, enquanto o modelo andragógico a aprendizagem é autodirigida. Os modelos se diferem: na necessidade do aluno aprender; no autoconceito; na experiência existente; na prontidão para aprender; na orientação para aprender e na motivação.

Pesquisadores relatam que, mesmo sem saber, os alunos acabam utilizando os dois modelos e que no modelo tradicional os instrutores tem uma sobrecarga de atividades devido a necessidade de auxiliarem dos alunos nas diversas atividades e que o material disponibilizado na Web pode ajudar.

Um protótipo foi desenvolvido para disponibilização do conteúdo na Web com um projeto de pesquisa pré-existente de introdução a programação com os objetivos de ajudar alunos de outros cursos com dificuldade e tutoria on-line ajudando a melhorar a fase 2 dos 4 estágios de aprendizagem. Os quatro estágios são: não sabe que não sabe; sabe que não sabe; sabe que sabe e não sabe que sabe.

Três fases foram definidas para a metodologia, A- Identificar as orientações de aprendizagem pretendida; B- Desenvolver o protótipo de aprendizagem on-line e C- Pesquisa experimental.

Como resultados e discussões, vamos analisar as três fases metodológicas:

A- Identificar as orientações de aprendizagem pretendidas:

Uma pesquisa com 262 alunos dos 469 inscritos em que se tentou identificar as orientações. Foram 48 itens na pesquisa, 24 suposições sobre pedagogia e 24 suposições sobre andragogia. Alguns do itens do pressuposto da pedagogia foram: "O palestrante ou professor é expediente o suficiente para determinar quais assuntos que me adequar" e "Eu prefiro o palestrante para me mostrar passo a passo soluções nos temas ensinados". Para o pressuposto da andragogia: "Eu que uma explicação dos benefícios deste assunto que é útil e pode ser usada ao mesmo tempo no mundo real" e "Gostaria de ir para a informação extra se eu estou interessado em algo afim de conhecê-lo ainda mais". Os resultados mostram que os alunos estavam no estágio 2 (93,9% - 246 alunos) – alta pedagogia e alta andragogia, com as seguintes características :

B- Desenvolver o protótipo de aprendizagem on-line.

Após uma revisão da literatura foi desenvolvido um protótipo de um ambiente de aprendizagem on-line individualizada com base na pedagogia e andragogia como seu modelo fundacional. Os alunos em estágio 2 precisam de objetivos, estratégias de aprendizagem e avaliação a ser definido pelo professor. O ambiente de aprendizagem on-line permite que o aluno a explorar os módulos de aprendizagem facilmente. Os alunos podem seguir para o próximo módulo, pois são aprendizes independentes com um nível moderado de autodirecionamento.

C- Pesquisa experimental.

Para medir o conhecimento após a tutoria on-line, foi utilizado um pré-projeto experimental de um grupo. Em 5 semanas (2h de aula teórica e 1h de laboratório), os alunos realizaram um pré-teste (início) e um pós-teste (final) e depois é feita um comparação entre os dois testes.

Na sexta semana, quatro alunos foram escolhidos aleatoriamente para uma entrevista de 10 minutos com o objetivo de descobrir informações mais concretas sobre o estudo. As respostas foram positivas no sentido que os alunos concordam que por usar o projeto experimental baseado na Web como mais uma referência em seus estudos. Ainda que pode melhorar o desempenho pois os exercícios previstos envolvem o pensamento crítico e são muito eficazes para ser usados antes de um teste ou exame, podendo saber as respostas imediatamente.

Os alunos ainda precisam de orientações de seus professores, não estando preparados a responsabilidade de planejar o seu próprio processo de aprendizagem. Na comparação do método tradicional (pedagógico) com o método de aprendizagem autônoma (andragógico) com o auxílio

de tutoria Web podemos concluir que os alunos usam os dois métodos no dia a dia, precisando de um acompanhamento de seus instrutores.

ARTIGO 05 O artigo trata de um curso de pensamento computacional para que alunos tenham habilidades analíticas, pensamento crítico para resolução de problemas lógicos e matemáticos. Este curso é dividido em duas partes: aulas teóricas/exercícios em sala de aula e aulas práticas em laboratórios. Os alunos podem fazer uma aula de aula teórica (2 créditos) ou uma aula de aula teórica e aula em laboratório (3 créditos). O curso é ministrado em 2 horas de aula teórica e 3 horas de laboratório por semana.

Nas aulas teóricas os alunos foram divididos em grupos e nos primeiros 15 minutos recebe as instruções e depois exercícios sobre os tópicos (algoritmo, tipo de dados e estrutura, abstração, simulação, iteração, recursividade). Além disso recebe tarefas adicional para reforço.

Os alunos não eram obrigados a fazer as aulas de laboratório, mas 74% fizeram aulas práticas e laboratório. Os alunos usaram o Kit LEGO Mindstorms para realizar os exercícios e depois aplicar os conteúdos ensinados nas aulas teóricas. Primeiro os alunos utilizas o ambiente gráfico de programação e depois refazem os mesmos códigos em Java (LEJOS). Os alunos também recebem trabalhos para programar os Robôs LEGO, tal como: o robô tem que sentir um objeto a uma determinada distância. Se a distância for segura o robô deve recuar, mas se o "intruso" se aproximar muito, o robô deve atacar.

Para avaliar o curso, os autores fizeram um teste no primeiro dia de aula (pré-teste) e outro no último dia (pós-teste) utilizando o método WASI para as turmas CS0 e FESP 095 (para testar e posteriormente comparar os resultados). Os testes WASI contêm perguntas que são classificadas em várias categorias: raciocínio verbal, instruções sequenciais, analogias, frases de relacionamento, análise de tendências e padrões, e problemas de matemática. Nenhum desses problemas requer habilidades matemáticas além de álgebra básica, embora a maioria dos problemas requerem habilidades lógicas ou analíticas e atenção aos detalhes. Os alunos das duas turmas tem as mesmas características. No pré-teste são aplicados 38 problemas aplicadas no primeiro dia de aula sem que os alunos fossem avisados. Isso também acontece no pós-teste. No pós-teste são 37 problemas semelhantes aos do pré-teste com maior grau de dificuldade. Nem todos os alunos fizeram os dois testes e para serem considerados nos resultados, os alunos teriam que fazer o pré e pós testes.

Na turma de **CS0**, 35 alunos se inscreveram na disciplina, mas somente 22 concluíram todo o curso. Como resultado, no pré-teste, o intervalo foi de 31,6% a 89,5%, com uma média de 60,3% e no pós-teste, o intervalo foi de 51,4% a 100%, com uma média de 78,4%. A melhoria na média em pontos percentuais foi de 18,1%. Na turma de **FESP**, 52 alunos se inscreveram

e 29 concluíram a disciplina. No resultado no pré-teste, o intervalo foi de 21,1% a 81,6% e no pós-teste, de 32,4% a 81,1%. A melhoria na média em pontos percentuais foi de 4%.

Foi mencionado anteriormente que nem todos os alunos participaram das aulas teóricas e das aulas em laboratório. Aproximadamente 26% dos alunos fizeram apenas as aulas práticas, 6 alunos e 16 levaram ambos, e tiveram avaliação superior em relação aos alunos que somente participaram das aulas teóricas

Os autores concluíram que após análise os resultados dão indícios promissores que o CS0, pensamento computacional, são positivos e impactam significativamente as habilidades de resolução de problemas analíticos. Isto sugere que o resultado do ensino de pensamento computacional melhora a capacidade analítica dos estudantes em geral.

ARTIGO 06 O artigo compara a eficácia de livros eletrônicos, onde o conteúdo é estático, com Web interativa, onde o conteúdo é dinâmico, para ensinar linguagem de programação C++. Basicamente o conteúdo estático é a disponibilização de livros eletrônicos e o conteúdo interativo consiste na disponibilização de animações, perguntas com respostas dinâmicas e além disso gravações de atividades de alunos em laboratório.

Os participantes do experimento eram alunos calouros, sendo que 79% eram da Engenharia. Estes alunos não sabiam o porque do experimento e ficaram das 9h até as 16h em laboratório executando atividades em linguagem de programação C++ e recebiam um crédito extra na disciplina. No projeto, 136 alunos foram avaliados em relação a quantidade de aprendizado e engajamento ou envolvimento nas atividades.

O material utilizado em laboratório foram o livro eletrônico e a página na Web. No experimento os alunos não eram identificados e tinham que seguir 7 passos, não podendo avançar se não concluísse o passo anterior. Estes passos estavam distribuídos em um questionário como segue:

- passo 1 – o alunos recebiam um identificador exclusivo;
- passo 2 – identificação: período na faculdade; experiência em programação; experiência com livros didáticos digitais;
- passo 3 – pré-lição com 4 perguntas de múltipla escolha;
- passo 4 – lição aleatoriamente escolhida em C++ disponibilizada tanto em conteúdo estático como conteúdo interativo (Web). Os alunos utilização o conteúdo e mais uma ferramenta de desenvolvimento no ambiente **Zyante**;
- passo 5 – pós-lição com 11 perguntas (8 de múltipla escolha e 3 aberta);
- passo 6 – pesquisa de acompanhamento em relação se o participante gostou ou não;

- passo 7 – agradecimento.

Todo o experimento foi conduzido por dois pesquisadores que recepcionavam o alunos e controlavam o tempo para a execução dos passos.

Os resultados foram considerados satisfatórios, pois os alunos tiveram um progresso no aprendizado e se envolveram mais. Analisando os alunos que declararam ser menos preparados, com menor conhecimento em programação, 64% melhoram seus resultados com a Web Interativa. Em relação aos acertos, no pré e pós lição, segue comparação na Tabela 2.7:

Tabela 2.7: Comparação dos acertos no pré e pós lição

Pesquisa	Livro Eletrônico	Web Interativa
Pré-lição (acertos)	1,8	2,2
Pós-lição (acertos)	7,3	8,6

Concluindo, o conteúdo Web foi mais eficaz em 16%, os alunos menos preparados melhoraram seu conhecimento em programação em 64% na media e todos os participantes ficaram mais tempo envolvidos com as atividades, aumentando o tempo de utilização do Livro Eletrônico de 9,4 minutos, para 17,5 minutos utilizando a Web Interativa.

ARTIGO 07 O artigo mostra o desafio de ensinar MATLAB e JAVA, no currículo comum do curso, com a mesma profundidade para alunos de engenharia da Universidade de Cornell. O curso é oferecido pelo departamento de Computação e tem o objetivo de ensinar os conceitos básicos de programação Java e ensinar uma ferramenta prática para a área da engenharia, o Matlab.

A Tabela 2.8 abaixo mostra a distribuição do conteúdo dos dois cursos. Obs: o conteúdo de Matlab esta sombreado na tabela.

Para avaliar os cursos, três questões serão analisadas:

1. Matlab motivou os alunos?
2. O alunos entenderam o objetivo do curso?
3. Os alunos CS100M saíram melhores preparados que os alunos de CS100J?

Para a primeira questão, os autores consideram que não motivou muito, pois ficou a mesma população de estudantes antes da implantação do novo curso. A segunda foi considerado que os alunos não entenderam o objetivo de CS100M, pois tinham receio que não fosse suficiente para a próxima disciplina de programação, CS211. Por fim, na terceira questão os de CS100M tiveram um percentual maior de notas “A”, mas não parece que os dois cursos ajudaram ou prejudicaram o desempenho global dos alunos.

Tabela 2.8: Distribuição do conteúdo dos dois cursos

Semana	CS100M	CS100J
1	Introdução à resolução de problemas e algoritmos, cessão, elementar funções	Introdução à resolução de problemas e algoritmos, cessão, entrada / saída
2	Ramificação, scripts	Ramificação, entrada / saída
3	Iteração	Iteração
4	1 matriz tridimensional	Iteração
5	Funções, organização do programa	Classes, objetos, métodos
6	Arquivo de entrada / saída, strings, gráficos	Classes, objetos, métodos
7	Matriz de 2 dimensões, gráficos	Classes, objetos, métodos, strings
8	Fundamentos Java, ramificação	Strings
9	Iteração	1 matriz tridimensional
10	Classes, objetos, métodos	Triagem, busca linear
11	1 matriz tridimensional, a triagem	MATLAB matriz 1-dimensional, gráficos
12	Classes, objetos, métodos	2-d matrizes
13	Herança	MATLAB matriz de 2 dimensões, funções
14	Strings 2-dimensional da matriz	Herança

Concluindo, a necessidade de um currículo único fez com que a partir de 2000 o novo curso com conceitos de Matlab fosse introduzido visando a aproximar os conceitos com o ferramentas práticas para a engenharia. Aparentemente o uso de Matlab não ajuda e nem prejudica a desenvolvimento dos alunos, apesar do aumento de notas "A" em CS100M.

ARTIGO 08 O artigo relata a experiência de um curso de introdução a computação para cursos de engenharia, com a disponibilização dos vídeos dos slides da aulas teóricas (material online), tentando suprir a falta de aulas teóricas. Após os alunos assistirem os vídeos eles vão ao laboratório fazer os exercícios práticos baseados na resolução de problemas (PBL) com a supervisão de instrutores.

Outra mudança, após comentários de professores e alunos, foi a mudança da linguagem de programação C para Matlab em de 2009 com 2 dias de aulas teóricas e 2 dias de laboratório e em 2010 veio a versão hibrida com a disponibilização do material online para o autoestudo (aulas teóricas gravadas, discussão, exercícios e perguntas antes de cada aula de laboratório).

A estrutura do curso constava de modulo online com o conteúdo idêntico ao modo tradicional, com questões de múltipla escolha e que tem que ser respondida antes da próxima aula. Também haviam perguntas de autoavaliação. As práticas de laboratório reforçavam o conteúdo online onde os vários pequenos programas são implementados, o alunos poderiam acompanhar o seu

código sendo executado. Os instrutores acompanhavam todas as atividades de laboratório e aplicavam as avaliação dos trabalhos.

O curso era avaliado com uma pesquisa sobre o conhecimento do alunos em relação ao sistema operacional, hardware, área de transferência e ainda sobre programação, 44% declaram conhecimento. A outra forma de avaliação eram perguntas sobre a satisfação com a experiência híbrida (material online), os alunos se mostraram satisfeitos.

O tempo de tutoria era de 5 noites, das 19 as 22 horas, com dois tutores. Conforme a necessidades mais tutores eram disponibilizados. Estes tutores eram alunos experientes. Um grupo fechado no Facebook foi criado para ajudar a comunicação entre alunos e tutores, acontecendo troca de experiência com perguntas e respostas referente ao conteúdo previsto. Isso ajudou a reduzir a falta de interação, pois alunos ficavam menos tímidos.

Os exames de avaliação eram feitos por uma semana inteira. Antes aconteciam duas aulas teóricas de revisão e duas sessões de laboratório, uma com um exame tipo simulado e outra com o exame real. A critério dos professores, para compor a nota, perguntas de múltipla escolha valiam entre 10 e 30% e a programação entre 10 a 90%. O alunos podiam utilizar todo o material mas não podia haver comunicação entre eles.

Concluindo, o curso híbrido ajudou os alunos na construção com conhecimento de programação. Projetos baseados na Web ajuda os alunos a administrar seu tempo de estudo. Com o material online os alunos necessitam de menos tempo de tutoria.

ARTIGO 09 Segundo os autores, o novo curso feito especialmente para Engenheiros foi baseado no livro "Introdução a Programação Científica" e por um site com o material online. As aulas de introdução a programação tem dois problemas: ignoram o contexto e são planejadas para alunos de Ciências da Computação. Para tentar resolver estes dois problemas é que este novo curso para engenheiros foi estruturado.

O curso tem quatro características: utiliza como linguagem de programação o Maple em combinação com C; ensina os conceitos de programação em paralelo com a metodologia de programação baseada em solução de problemas; os exercícios são relacionados a Ciências e Engenharia; não se utiliza somente das aulas teóricas, mas também com o material disponibilizado na Web. No cenário do curso estudado pelos autores, o método de resolução de problemas computacionais consiste em 5 etapas:

1. Entender o problema e saber exatamente o que fazer para resolver;
2. Criar um modelo matemático;
3. Criar o método de resolução do modelo matemático;

4. Aplicar o método na linguagem de programação C;
5. Avaliar a solução.

Para motivar os alunos na resolução de problemas computacionais é necessário trazer o problema para a realidade do aluno ou do curso, por exemplo: média da população; movimento de um projétil; controlar o joelho, tornozelo e quadril de um robô; etc.

Para motivar os alunos, o material online do curso foi disponibilizado na Web com um hipertexto com o conteúdo idêntico ao do livro e ainda animações no Maple para demonstrar exercícios e problemas resolvidos em sala. O curso foi ministrado semanalmente, com 2 aulas teóricas e 2 aulas práticas no laboratório com conteúdo bem detalhado que era discutir o problema, modelo e método e ao final apresentar a implementação.

Em 1996 e 1997 desenvolveram a terceira geração do material online com applets Java para ilustrar melhor os conceitos passadas em aulas teóricas e para trazer problemas mais relacionados com o curso do aluno e o resultado foi satisfatório.

ARTIGO 10 O artigo mostra o uso do paradigma do paraquedas para ensinar programação e deixar o ensino aprendizagem mais adequado para as gerações atuais. A ideia é mostrar aos alunos como analisar um sistema numa visão global, mapear este sistema e depois refinando ou descendo de níveis para entender os detalhes deste sistema. Um sistema pode ser dividido em partes, aumentado assim a precisão da análise e conseguimos observar o estado e eventos deste sistema.

O curso foi dividido em 16 semanas com 32 sessões de 2 horas com a metodologia e conteúdo consistindo da divisão e entendimento das partes do sistema computacional com exemplos reais para que os alunos entendam e essência da programação. As partes são: Sistema, Observação, Estado, Condição, Abstração, Recursão, Iteração e Abstração de Dados.

- **Sistema:** é passado ao alunos o conceito de sistema em si e que os sistemas podem ser divididos e modelados obtendo assim os elementos que compõe este sistema e a interação entre esses elementos. O sistema de computadores são sistema reais. (1 sessão)
- **Observação:** os alunos conseguem identificar os elementos estáticos e os que mudam. Os elementos estáticos são as constantes e os que mudam são as variáveis. Os alunos tem contato com códigos básicos em Python. (1 ou 2 sessões)
- **Estado:** descendo mais o nível, o aluno percebe que os valores das variáveis mudam, verificando assim a mudança de estado destas variáveis. Aprendem o conceito de algoritmo e entradas e saídas. (2 sessões)

- **Condição:** os alunos tem contato com estrutura de controle, lógica de programação, estruturas relacionais, tabela verdade, etc. (4 semanas)
- **Abstração:** alunos resolvem problemas mais complexos e com código maiores e aprendem a reduzir o tamanho dos códigos colocando várias operações numa única. Os problemas são ligados a Engenharia, trazendo mais próximo com a realidade. Aqui acontece a primeira avaliação em duas partes: uma com computador e outra sem computador.
- **Recursão:** com exemplos reais os alunos aprendem a chamada de funções e repetição. (4 a 6 sessões)
- **Iteração:** verificando a mudança de estado das variáveis, alunos aprender laços e comando WHILE e acontece a segunda avaliação. (várias sessões)
- **Abstração de Dados:** alunos tem que desenvolver um código para calcular a médias de várias turmas, tendo assim contato com listas e acontece a terceira avaliação.

Aplicando o estudo de caso, os autores observaram que após dois anos de realização do curso, os alunos são capazes de modelar um sistema e conseguem programar com a visão de resolução de problemas reais da engenharia utilizando Python. Concluindo, os alunos aumentaram a motivação em 70% e as notas em 30%.

ARTIGO 11 O artigo relata o curso prático para calouros de Engenharia Elétrica na Alemanha com integração de Matlab e LEGO Mindstorms. São 309 alunos, 60 tutores ou supervisores em 23 Institutos. O curso é focado em aprendizagem tripla: métodos matemáticos, programação Matlab e Engenharia prática, onde os alunos transferiram os fundamentos para algoritmos em Matlab afim de controlar o robô LEGO com o objetivo de aprendizagem bem definido para exercícios e atividades da Engenharia Elétrica. Assim os alunos adquirem competências transversais com conhecimentos reais da engenharia, melhoram a programa em Matlab, aumentam a motivação e permite um processo de aprendizagem de programação em pares.

Os alunos foram submetidos a um teste para medir o conhecimento e dois meses antes do início do curso os alunos tem introdução ao Matlab com programas curtos para obterem conhecimento mínimo da linguagem.

O curso foi inicial em 2007 em tempo integral em 8 dias, das 8h até as 16h. Nos laboratórios tinham 150 estações de trabalho e 100 Kits LEGO Mindstorms.

O curso tem atividades práticas em grupos divididas em três em : A- Exercícios Básicos; B- Tarefas Individuais; e C- Apresentação. Na atividade de Exercícios Básicos os grupos tinham cinco exercícios: 1º – em 1 horas, os alunos tinham que montar um robô e fazer seguir uma linha preta numa folha ou superfície branca. Isso deixa os alunos muito animados; 2º – os alunos

testavam a comunicação Bluetooth e tinham que ler o sensor de toque; 3º – programação mais avançada com loop de repetição, condicionais e matrizes (controle de semáforo); 4º – utilizando os motores servos os alunos faziam medição do sensor de rotação interna e trabalhavam com diferentes relações de transmissão de engrenagens; 5º – utilização do sensor de luminosidade e sensor de cor em comparação ao olho humano e 6º – desenvolver um robô que explora o local com o sensor de ultra-som. Nas tarefas individuais, na segunda parte do laboratório, os alunos tinham desafios práticos mais complexos. Em equipes de quatro alunos eles poderiam desenvolver suas próprias ideias ou algumas atividades predefinidas como: saída de labirinto; agarrar e ordenar bolas coloridas e scanner para digitalizar imagens em escala de cinza(tratamento de imagem). No último dia do projeto, as equipes de alunos apresentam os seus resultados e construções robô individuais para os outros participantes e os supervisores em 15 minutos.

Para avaliação dos projetos os alunos tinham que fazer todas as atividades e os supervisores atribuíam pontos de crédito. Dos 309 alunos, 93% (287) conseguiram sucesso no projeto/curso. Além disso os alunos eram convidados a fazer uma pesquisa/avaliação anônima e voluntária, onde 131 estudantes responderam 38 perguntas sobre conceito geral do projeto, exercícios específicos e melhorias pessoais alcançadas. Os alunos comentaram sobre a falta de kits, pois tinham que dividir com outras equipes. No geral os alunos os alunos se sentiram motivados durante os 8 dias do curso. Além disse os supervisores relataram uma integração da equipe e entres as equipes em relação a trabalho em pares quando os estudantes partilharam recursos e estavam engajados nos debates.

ARTIGO 12 O artigo mostra a reformulação da disciplina CS50, em Harvard. A disciplina tinha bons materiais, bons professores, mas tinha problemas na percepção e no design. CS50 era considerada uma das disciplinas mais difíceis e mais rigorosa da grade. A rigorosidade foi mantida na reformulação, mas a ideia era deixar a disciplina mais amigável, mais próxima dos alunos, acompanhando a tendências tecnológicas que são mais familiares a estes alunos. Os números mostram que, após a reformulação, os números de matrículas aumentaram, chegando a 338 em 2009, números próximo ao de 1996 que foi 386 (o mais alto desde 1989).

Dos alunos matriculados, 72% nunca fizeram a disciplina, dos quais 34% se sentiam desconfortáveis, 14% confortáveis e para 52% indiferentes a situação. Os alunos se reuniam 2 vezes por semana para as aulas teóricas e ainda eles ficavam entre 10 e 20 horas além das aulas teóricas estudando. No início os novos alunos ficavam com um aluno que já havia feito a disciplina, um companheiro de ensino.

O curso foi dividido em 10 semanas com aulas teóricas, orientações, tutoria virtual, sessões de vídeos, clips no YouTube, os alunos podiam fazer perguntas anonimamente. O curso focou problemas ligados a área do curso do aluno, tornando-se mais amigável e com a rigidez necessária.

Segue o currículo do novo curso com conteúdos de cada semana:

- Semana 0 – declarações, variáveis, expressões booleanas, condicionais, loops.
- Semana 1 – Linux. Linguagem C – padrão de entrada e saída, funções e bibliotecas.
- Semana 2 – Tópicos avançados em C, matrizes. Exemplos do mundo real: criptografia, para deixar mais atraente para os alunos.
- Semana 3 – pesquisa binária e classificação. Depuração dos algoritmos para que os alunos vejam as linhas de código.
- Semana 4 – gestão de memória em C, ponteiro. Revisão nas semanas seguintes.
- Semana 5 – arquivos de entrada e saída. Ler e gravar em disco. Listas.
- Semana 6 – Estruturas de dados mais sofisticadas, tabela hash, problemas interessantes.
- Semana 7 – Programação Web, XHTML, PHP e SQL. Projeto final na Web.
- Semana 8 – Programação Web. Introdução ao DOM, JavaScript.
- Semana 9 – Compilar, montar e código.

Na definição dos problemas que as lições a serem estudadas são apontadas a cada semana.

- Problema 0 – projeto definido pelos alunos.
- Problema 1 – Linux. Programas em C: validação de ISBN ou de cartão de crédito.
- Problema 2 – mundo real: criptografia.
- Problema 3 – código para jogo dos Quinze (http://www.testonline.com.br/quinze.htm).
- Problema 4 – código Sudoku – 600 linhas.
- Problema 5 – código em C para recuperar imagens deletadas de um máquina digital.
- Problema 6 – competição entre os alunos – corretor ortográfico.
- Problema 7 – implementar PHP e SQL – compra e venda.
- Problema 8 – implementar “mashup” usando google maps.

Os resultados mostram que as matrículas ou inscrições em CS50 aumentaram 156% em três anos. Concluindo, a reestruturação de CS50 obteve sucesso em relação ao aumente de inscrições e que, apesar de ser um curso para todas as áreas, inclusive Engenharias, o interesse dos alunos de Computação aumentou.

ARTIGO 13 O artigo mostra como ensinar introdução a computação através de um abordagem baseada em projeto de aprendizagem colaborativa para alunos de Engenharia Eletrotécnica. O

curso tinha aulas teóricas e aulas práticas em laboratório, utilizando Python como linguagem de programação e que é de fácil entendimento pois são alunos com pouca experiência. O curso é dividido em dois tipos: no primeiro os alunos são separados em grupos de 2 ou 3, que interagem de forma síncrona no laboratório por duas horas, buscando negociar e construir representações esquemáticas de problemas algorítmicos simples; no segundo tipo os alunos formam grupos de 4 ou 5 alunos por 5 semanas para desenvolver um projeto em Python.

O laboratório é composto por 40 estações de trabalho com acesso a internet e ao Moodle, onde 3 assistentes ou instrutores acompanham as construção de algoritmos referentes a exercícios feitos em sala de aula. No laboratório os alunos são separados em grupos de 2 ou 3 alunos distantes entre si para evitar a comunicação. Eles se comunicam por Groupware utilizando o software Sinergo que auxilia na representação esquemática apresentando diagramas, mapas conceituais, etc. Os projetos eram avaliados pela qualidade da resolução e pelo esforço colaborativo.

Algumas limitações foram descobertos nesta abordagem. Os alunos sentiam falta de uma definição clara da melhor prática colaborativa; os alunos experientes reclamavam da conversa dos outros alunos dentro do grupo; os instrutores reclamavam sobre a falta de instrução de como calcular o esforço na colaboração. Todos os problemas foram resolvidos nos anos seguintes e concluíram que os professores e instrutores também tem que ser treinados para o experimento da atividade colaborativa e ainda que na formação dos grupos tem que se considerar as habilidades e conhecimentos dos estudantes.

Foi realizada uma avaliação anônima onde 164 alunos responderam se a experiência foi positiva ou negativa. Os resultados apontaram que 60% consideraram positiva ou muito positiva e 13% negativa. Também haviam questões abertas para sugestões e melhorias no projeto, onde 93 alunos responderam(56%) com comentários positivos (+) e negativos (-), como segue na Tabela 2.9. 20 alunos encontraram dificuldade nas tarefas ou atividades e 18 disseram que o tempo foi limitado.

Tabela 2.9: Análise dos resultados

Dificuldade nas atividades (-)	20
Tempo alocado limitado (-)	18
Suporte limitado (-)	14
Experiência positiva (+)	9
Benefícios do trabalho em equipe (+)	7
Livre escolha de parceiros (-)	6
Dificuldades em colaboração (-)	6
Assunto interessante (+)	4
Livre escolha de assunto (-)	3

Uma comparação foi realizada entre 2008-2009 (173 alunos) e 2009-2010(165 alunos). Nos dois anos as características do curso eram as mesmas em termos de currículo, corpo docente, material. A única diferença foi a implantação do projeto. Foram comparados na Tabela 2.10, mostrando que com o projeto, em 2009, um aumento em todas as questões. Por exemplo: Nível das aulas teóricas (satisfatório) aumentou em 2009(com projeto) para 2,96 em relação a 2008(sem projeto) que foi de 2,63.

Tabela 2.10: Comparação: sem projeto X com projeto

Questão	2008-2009 (sem projeto)	2009-2010 (com projeto)
Nível das aulas teóricas (satisfatório)	2,63	2,96
Atividade de laboratório	2,85	3,05
Qual curso mais interessante (computação)	1,26	1,86

Ainda foi realizado um estudo sobre os efeitos da aprendizagem do curso onde um teste em Python foi aplicado. Entre os problemas havia um que foi pedido nos dois anos comparados. O valor médio melhorou no ano com projeto, mas não foi tão significativa (p=0,281), mas os alunos progrediram em termos de suas habilidades de programação, como segue na Tabela 2.11.

Tabela 2.11: Desempenho no exame final (problema de programação)

ANO	**ALUNOS**	**MÉDIA**	**DESVIO PADRÃO**
2008 - 2009	179	4,711	2,991
2009 - 2010	164	5,065	3,094

Concluindo, os autores apontam que a experiência foi satisfatória para alunos e instrutores. A abordagem de aprendizagem colaborativa parece melhorar as habilidades sociais e de comunicação. Mas isso não ocorreu no grau esperado, houve a necessidade de atividades de orientação através de exemplos de boas práticas de trabalho colaborativo.

2.3.2. Análise e síntese dos resultados

Foram sintetizadas as linguagens e métodos evidenciados na revisão sistemática, conforme apresentado na Tabela 2.12 e Tabela 2.13.

Em relação às linguagens de programação, as linguagens utilizadas foram: Lego, Matlab, Java, Python, Maple e Excel. O LEGO foi utilizado por quatro autores (LIANG *et al.*, 2013; DYNE; BRAUN, 2014; BEHRENS *et al.*, 2010), sendo que um deles considerou sua utilização em trabalho futuro (SARRIA; GERALDO, 2009). Linguagens voltadas à matemática e mais próximas ao universo de Engenharia, tal como Matlab e Maple, também foram amplamente utilizadas em detrimento a linguagens mais tradicionais na Computação, como Java e Python.

Tabela 2.12: Linguagem de programação na Intervenção

#	ano	Lego	Matlab	Java	Python	Maple	Excel
01	2013	X					
02	2003						X
03	2014		X				X
04	2014						
05	2014	X					
06	2014					X	
07	2004		X	X			
08	2012		X			X	
09	1996					X	
10	2009	Trab.Futuro					
11	2010	X					
12	2010	Reformulando CS50					
13	2010				X		

Os métodos utilizados foram: disponibilizar material na Web, Aprendizagem Baseada em Problemas (PBL), Aprendizagem Colaborativa e Aprendizagem Autodirigida-Andragogia. A PBL, método foco neste trabalho que mestrado, foi aplicado por três autores (MAHADEVAN-JANSEN *et al.*, 2003; SUN *et al.*, 2012; ZACHARY, 1996).

Tabela 2.13: Métodos utilizados na Intervenção

#	ano	Web	PBL	Aprendizagem Colaborativa	Aprendizagem Autodirigida-Andragogia
01	2013				
02	2003		X		
03	2014				
04	2014	X			X
05	2014				
06	2014	X			
07	2004				
08	2012		X		
09	1996		X		
10	2009				
11	2010				
12	2010				
13	2010			X	

2.4. Considerações finais

Computação é um instrumento utilizado em Engenharia para o desenvolvimento de raciocínio lógico e computacional necessários para resolver problemas complexos e também é requisito em cursos de graduação em Engenharia. Mas o ensino e aprendizado de Computação não é trivial e apresentam números elevados de reprovação e afetam a retenção de alunos nos cursos. Considerando este cenário, este trabalho fez um levantamento sobre as disciplinas de programação nos semestres iniciais dos cursos de Engenharia Ambiental, Engenharia de Alimentos e Engenharia Civil de nossa instituição com questionários específicos para alunos, dos professores e dos coordenadores.

Com a perspectiva destas entidades constatamos uma certa desarmonia com os planos de ensinos. Focando na importância, existem opiniões diversas e antagônicas, identificamos percentuais e relatos considerando muito importante e nada importante. Ainda muitas sugestões que nos leva a concluir que a disciplina introdutória à Computação é necessária para os cursos de Engenharias (Ambiental, Alimentos e Civil), mas existe a necessidade de melhorar a forma de estimular ou conscientizar os alunos desta importância para o seu curso. Considerando as atividades de ensino, pesquisa e extensão, a disciplina de Computação é relevante com um percentual maior no Ensino. Pontuando os fatores que afetam o ensino de Computação positivamente são a criatividade e raciocínio lógico; e negativamente é o comportamento e interesse dos alunos.

Alguns resultados mostraram que para os alunos a disciplina é importante, apesar de alguns considerarem nada importante; que coordenadores e professores do departamento de computação consideram a disciplina extremamente importante; que professores de outros departamentos necessitam do acréscimo de conteúdos específicos para suas disciplinas.

Capítulo 3

PBL e objetos de aprendizagem para ensino de Computação para Engenharia

Os estudos de Mahadevan-Jansen *et al.* (2003), Sun *et al.* (2012) e Zachary (1996) apontam para alternativas de novos processos de ensino aprendizagem que buscam deixar a disciplina mais interessante, fazendo com que os alunos não se desanimem e se engajem cada vez mais em seu curso de graduação. Uma destas alternativas é o PBL – *Problem Based Learning* ou ABP – Aprendizagem baseada em problemas, e que "apesar de constituir uma das maiores promessas para a revitalização dos métodos pedagógicos no Ensino Superior, ainda é pouco difundida" (GIL, 2008).

Em virtude dos resultados relatados no capítulo anterior, PBL apresenta-se como uma possível solução. Entretanto, é necessário sistematizar sua aplicação, logo a necessidade de objetos de aprendizagem que permitam a representação de elementos de PBL e que apoiem o seu uso.

Os métodos de desenvolvimento de OA's serão apresentados na Seção 3.2.1. Na Seção 3.1 será apresentada a aprendizagem Baseada em Problemas (PBL), que é uma abordagem onde os alunos tem o objetivo de resolver um problema.

3.1. PBL

A aprendizagem Baseada em Problemas (PBL) é uma abordagem onde os alunos tem o objetivo de resolver um problema. O aluno deixa de ser o receptor do conhecimento passado pelo professor se torna o condutor principal do seu próprio aprendizado. Para Gil (2015), o professor também tem um papel diferenciado do modo tradicional de ensinar e se torna um facilitador para que seus alunos cumpram as etapas de resolução do problema apresentado e construa o seu conhecimento baseado em sua experiência e em novos conhecimentos pesquisados ou compartilhados pelos membros de sua equipe.

Para exemplificar alguns OA's, foi colocado na seção ANEXOS„„ cinco problemas apresentados do tutorial do link http://sites.ecomp.uefs.br/mip-20131/home/tutorial

A ideia principal da PBL é buscar que o próprio aluno construa seu conhecimento e que ele se interesse em resolver o problema antes de receber os conteúdos, mudando assim sua postura no aprender. Assim o aluno adquire uma autonomia e mostra que pode compreender determinados assuntos ou conhecimentos com sua própria visão, mas o professor acompanhada todo o processo para avaliar ou validar o conhecimento adquirido.

3.1.1. Histórico da PBL

Em 1960, a PBL passa a ser utilizada de forma estruturada na Faculdade de Medicina da McMaster University (Canadá) (RIBEIRO, 2005). Entre 1980 e 1990, passa a ser amplamente adotada nos cursos de Saúde, Engenharia, Educação, Administração de várias universidades no Estados Unidos. Em 1990, começa a ser utilizada no Brasil por cursos nas áreas de Medicina e Engenharia.

Embora a PBL seja utilizada a muito anos com sucesso, ainda é alvo de críticas por não ter uma base científica, uma vez que seus idealizadores não se basearam em nenhum teórico para fundamentar o método. Contudo, os princípios que formam a base da PBL possuem muita semelhança com as teorias de Ausubel, Piaget, Bruner, Dewey, entre outros (RIBEIRO, 2008).

3.1.2. Características da PBL

A abordagem da PBL tem início com a apresentação de um problema aos alunos, sem nenhuma informação e instrução sobre a solução do problema. O problema em si tem a finalidade de fazer o alunos, individualmente ou em grupo, estudar os assuntos relacionados analisando o problema e possíveis soluções. Quando os alunos identificam as questões importantes, eles realizam o estudo

individual antes de se reunir com o grupo para que todos os membros troquem informações e conhecimentos pesquisa e ainda os da sua experiência e vivência, para tentar resolver o problema (MAMEDE, 2001). Segundo Ribeiro (2008), a fase final da abordagem envolve a atividade de reflexão para que os alunos avaliem seu próprio progresso e também o dos outros membros do grupo.

Uma maneira de sistematizar a PBL é seguir os sete passos propostos pela Universidade de Maastricht Deelman e Hoeberigs (2009). O "Referencial de Maastricht" propõe que, ao receber a situação problema, o grupo busque solucioná-la seguindo sete etapas:

1. Leitura da situação problema e esclarecimento de termos desconhecidos;
2. Identificação do problema proposto pelo enunciado;
3. Discussão do problema e formulação de hipóteses para resolvê-lo;
4. Resumo das hipóteses;
5. Formulação dos objetivos de aprendizagem. Com base nos conhecimentos prévios são identificados os assuntos que devem ser estudados para a resolução do problema;
6. Estudo autônomo dos assuntos levantados no passo anterior;
7. Retorno ao grupo tutorial para discutir novamente o problema à luz dos novos conhecimentos adquiridos na fase de estudo autônomo.

Para Berbel (1998), os sete passos da PBL compreendem duas fases:

- Na primeira, a discussão é focada na identificação do problema, elaboração de hipóteses de solução e identificação de assuntos relevantes para a solução dos problemas;
- Na segunda, os conhecimentos prévios são confrontados com os conhecimentos científicos que o aluno busca de forma autônoma.

3.1.3. Experiências de PBL para Computação em Engenharia

A aplicabilidade da PBL como método de ensino de computação para cursos de Engenharia é feita através da implementação de estudos relacionados a problemas específicos de cada área da engenharia (MAHADEVAN-JANSEN *et al.*, 2003; SUN *et al.*, 2012; ZACHARY, 1996).

Por exemplo, o que foi citado no artigo de Zachary (1996), como segue:

> O método de resolução de problemas computacionais consiste em 5 etapas:
>
> 1. Entender o problema e saber exatamente o que fazer para resolver;
> 2. Criar um modelo matemático;
> 3. Criar o método de resolução do modelo matemático;

4. Aplicar o método na linguagem de programação C;
5. Avaliar a solução.

Em outro artigo, Sun *et al.* (2012) relata a experiência de um curso de introdução a computação para cursos de engenharia, com a disponibilização dos vídeos dos slides das palestras (material online), tentando suprir a falta de aulas teóricas. Após os alunos assistirem os vídeos eles vão ao laboratório fazer os exercícios práticos baseados na resolução de problemas (PBL) com a supervisão de instrutores.

3.2. Objetos de aprendizagem

Segundo Wiley (2000b), OA é "qualquer recurso digital que possa ser reutilizado para o suporte ao ensino". Esta definição será a considerada neste trabalho.

Os Objetos de Aprendizagem (OA) podem ser criados em qualquer mídia ou formato, podendo ser simples como uma animação ou uma apresentação de slides ou complexos como uma simulação, utilizando imagens, animações, arquivos de texto ou hipertexto, dentre outros. Deve ter um propósito educacional definido e poder ser aplicado em diversos contextos. Considerando que toda atividade de aprendizagem sempre se dá em um novo contexto, por mais similares que sejam a contextos anteriores, simplesmente porque o ator – aluno – é diferente (mesmo se for o mesmo aluno, porque seu comportamento em um tempo T é diferente daquele no tempo T). Ainda tem a característica de ser reutilizado em diferentes contextos ou disciplinas, na forma presencial ou à distância, sempre apoiado pela utilização de tecnologias atuais como: computadores, tablets, celulares, plataformas de ensino a distância, dentre outros (WILEY, 2000b).

A utilização e reutilização de um OA de forma independente necessita que os dados sejam armazenados em repositórios para serem comparados no futuros. Os metadados são utilizados para recuperar, reutilizar e combinar diferentes OA's na forma de módulos. Assim o princípio de um OA é permitir que seja fracionado e suas partes combinadas e reutilizadas em diferentes contextos e cenários de acordo com um projeto instrucional (WILEY, 2000a).

Os objetos de aprendizagem são recursos digitais utilizados para o ensino. Um OA pode ser usado em diferentes contextos e em diferentes ambientes virtuais de aprendizagem, interagindo com o aluno, promovendo a criatividade por meio de jogos, simulações, vídeos, etc., complementando o conteúdo da disciplina e tendo como característica a reusabilidade.

Spinelli (2005) acrescenta que estimula o desenvolvimento das capacidades pessoais do aluno. Audino (2012) complementa dizendo que:

> são recursos digitais dinâmicos, interativos e reutilizáveis em diferentes ambientes de aprendizagem elaborados a partir de uma base tecnológica. Desenvolvidos com fins educacionais, eles cobrem diversas modalidades de ensino: presencial, híbrida ou a distância; diversos campos de atuação: educação formal, corporativa ou informal; e, devem reunir várias características, como durabilidade, facilidade para atualização, flexibilidade, interoperabilidade, modularidade, portabilidade, entre outras. Eles ainda apresentam-se como unidades auto-consistentes de pequena extensão e fácil manipulação, passíveis de combinação com outros objetos educacionais ou qualquer outra mídia digital (vídeos, imagens, áudios, textos,gráficos, tabelas, tutoriais, aplicações, mapas, jogos educacionais, animações, infográficos, páginas web) por meio da hiperligação. Além disso, um objeto de aprendizagem pode ter usos variados, seu conteúdo pode ser alterado ou reagregado, e ainda ter sua interface e seu layout modificado para ser adaptado a outros módulos ou cursos. No âmbito técnico, eles são estruturas autocontidas em sua grande maioria, mas também contidas, e marcadas por identificadores denominados metadados.

3.2.1. Métodos e processos de desenvolvimento

O desenvolvimento de um OA é bastante complexo e envolve a participação de uma equipe composta por pedagogos, desenvolvedores, designers gráficos e especialistas de área. Eles devem interagir de modo a atingir os objetivos tanto tecnológicos quanto pedagógicos (BOND *et al.*, 2008). Portanto é necessário o uso de métodos para organizar o processo de desenvolvimento, a padronização e a comunicação entre os envolvidos. A aplicação de práticas sistemáticas é fundamental para a garantia da produtividade do processo de desenvolvimento e da qualidade dos objetos gerados (BARBOSA, 2004). Caso se use um método equivocado ou nem seja usado nenhum, o resultado pode ser insatisfatório e não atingir seus objetivos e nem possa ser reutilizado em outros cenários. Algumas métodos ou abordagens são genéricos para o desenvolvimento de conteúdo didático e executadas por profissionais da área da Educação, talvez deixando de lado a abordagem técnica de desenvolvimento de software e por consequência disso desconsiderando a qualidade do OA e possivelmente comprometa a reusabilidade.

Os OA's podem ser desenvolvidos de várias formas, sem seguir regras ou padrões. Contudo para garantir que os OA's sejam eficazes para o ensino aprendizagem e possam ser reutilizados de alguma forma, parcial ou integralmente, seu desenvolvimento deve ser sistematizado, sempre levando em consideração a importância de se utilizar modelos e métodos. De acordo com Kroll e Kruchten (2003), um produto de qualidade deve ter ausência de defeitos e, principalmente, deve atender aos propósitos desejados.

Silva (2012) comenta que, na literatura, são observadas duas gerações de métodos para o desenvolvimento de objetos de aprendizagem. A primeira adota a visão de material didático como um hipertexto. Posteriormente, a atenção dirige-se aos padrões definidos por organizações como *Advanced Distributed Learning* (ADL), *Aviation Industry CBT Committee* (AICC) e *IMS Global Learning Consortium* (IMS) e, principalmente, à questão de reusabilidade. Na segunda geração, as técnicas preservam o uso de especificações formais e ainda assume-se o caráter interdisciplinar da atividade de desenvolvimento, definindo-se processos.

3.2.2. LOD (*Learning Object Development*)

Portanto existe a necessidade do uso de uma metodologia de desenvolvimento de OA que leve em conta os preceitos pedagógicos e técnicos. No estudo de Silva *et al.* (2011) foi utilizado a LOD, uma abordagem para desenvolvimento de objetos de aprendizagem multimídias e interativos. Ela compreende um método dirigido a modelos, um processo-padrão e um processo específico ao desenvolvimento de objetos e aprendizagem para televisão digital. A LOD foi construída a partir da abordagem IMA-CID *(Integrated Modelling Approach – Conceptual, Instructional and Didactic)* (BARBOSA; MALDONADO, 2006a, 2011), utilizado para representar projetos instrucionais com interações dos usuários e entre as informações associadas ao objeto de aprendizagem, e do processão padrão SP-DEM *(Standard Process - Development of Educational Modules)* (BARBOSA *et al.*, 2003; BARBOSA; MALDONADO, 2006b).

O LOD consiste de um modelo de processo ou processo-padrão para desenvolvimento de OA's (SPLOD *(*Standard Process for Learning Object Development), que define a instância para desenvolvimento de um OA (LODP *(Learning Object Development Process)*; um método de desenvolvimento dirigido a modelos, denominado LODM *(Learning Object Development Method)*, compreendendo a modelagem conceitual, instrucional e de interação do objeto de aprendizagem; e um conjunto de ferramentas que estabelecem um protótipo de ambiente para o desenvolvimento de objetos de aprendizagem (LODE *Learning Object Development Environment)* considerando o método. Neste trabalho trataremos apenas do LODM, pois somente modelaremos de OA.

3.2.3. LODM *(Learning Object Development Method)*

O LODM *(Learning Object Development Method)* é um método de desenvolvimento de OA composto de três modelos: modelagem conceitual, modelagem instrucional e modelagem de interação (SILVA *et al.*, 2011). Este método foi criado com foco na utilização de um processo mais robusto de desenvolvimento de OA's que envolva equipes multidisciplinares que trabalham

de forma aberta para a criação de recursos educacionais livres. Também os modelos do LODM servem ao propósito da modelagem de interações em sintonia com o projeto instrucional e ainda a geração de objetos conforme as plataformas de saída desejadas. A seguir um breve relato de cada um dos modelos do LODM.

Modelagem Conceitual é construído a partir da técnica de mapeamento de conceitos e representa os objetos educacionais na forma de um modelo de conhecimento estruturado em **conceitos** e **proposições**. Considerando o OA, o modelo conceitual estabelece os principais requisitos educacionais e identifica os conceitos relevantes para a compreensão do domínio do conhecimento e a especificação da forma pela qual eles se relacionam (NOVAK, 1977).

Modelagem Instrucional é a parte do projeto instrucional relacionada com os conceitos do modelo conceitual e representa as atividades de aprendizagem e os conceitos associados, planejados de acordo com os objetivos educacionais identificados no modelo conceitual. Este modelo define a organização das informações necessárias para as execuções das atividades de aprendizagem e permite o uso de diferentes métodos instrucionais representado-os como elementos instrucionais. Estes elementos têm a função de explanação, exploração ou avaliação.

Modelagem de Interação complementa o modelo instrucional. Ele define elementos de interação e reorganiza os elementos do modelo instrucional, definindo as atividades de aprendizagem em função da sincronização entre os elementos do modelo. De acordo com a plataforma ou ambiente de execução, os dispositivos de interação utilizados e as próprias mídias, o conjunto de eventos possíveis é ampliado.

Segundo Silva (2012), os modelos são definidos de forma iterativa, apresentados na Figura 3.1. Inicia-se pelo modelo conceitual, construído a partir de outros materiais didáticos ou modelos conceituais. Os conceitos e proposições são organizados de acordo com métodos instrucionais e representados no modelo instrucional. O modelo de interação define a interação com o OA a partir da sua representação no modelo instrucional.

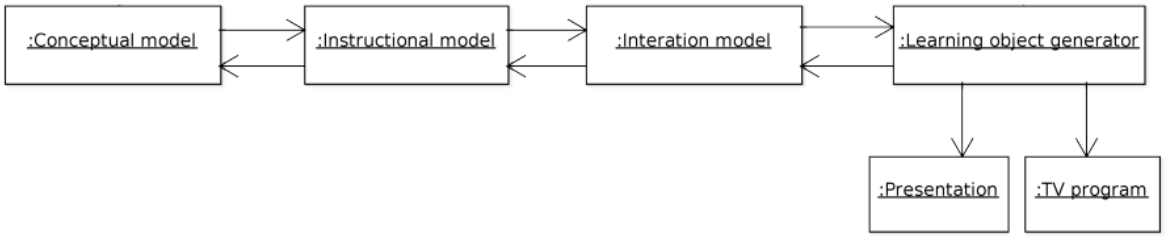

Figura 3.1: LODM: Visão geral do método. Fonte: Silva (2012)

Ao término da especificação das interações, escolhe-se a plataforma de execução (e.g., televisão digital), e geram-se objetos de aprendizagem a partir do modelo de interação. Caso satisfatórios, eles são utilizados na implementação da instrução.

3.3. Considerações finais

Os conceitos e características da PBL e experiências relatadas nos artigos de Mahadevan-Jansen *et al.* (2003), Sun *et al.* (2012) e Zachary (1996), ainda os métodos e processos de desenvolvimento de OA com a metodologia LOD proposta por Silva (2012), focando o método LODM, apresentados neste capítulo, nos encoraja a pesquisar sobre o assunto e cumprir o objetivo desta PDM que é propor um método de desenvolvimento de Objetos de Aprendizagem baseado em PBL, utilizando a abordagem LOD a ser aplicado em disciplinas de Computação nos cursos de Engenharia.

Capítulo

4

Proposta

Neste trabalho, o LOD será aplicado e o LODM será aperfeiçoado para o desenvolvimento de objetos de aprendizagem baseados em problemas para o ensino de introdução a programação nos cursos de Engenharia. As iterações serão executadas, preliminarmente, nos cursos de Engenharia do Campus Campo Mourão da Universidade Tecnológica Federal do Parana - UTFPR.

4.1. Método

O estudo será aplicado a alunos dos cursos de Engenharia com o método de estudo de caso conforme preconiza Yin (2001), sendo uma pesquisa empírica que investiga um fato contemporâneo dentro do seu contexto real.

O estudo será realizado de forma iterativa, com várias aplicações sucessivas e a cada aplicação sera feito algo a mais. O OA será desenvolvido baseado em PBL e utilizando o LOD e aperfeiçoando o LODM. As iterações serão aplicadas e turmas de primeiros e segundos períodos. Estes alunos serão convidados a fazer uma atividade que utilizará o OA criado.

4.2. Atividades

O levantamento e modelagem de características de PBL suprirá a deficiência de conteúdo encontrada na Seção 3.1, auxiliando assim no entendimento destas características e requisitos da PBL para aplicação e criação no LODM.

A criação de elementos no LODM para representar PBL será uma das atividades importantes para o entendimento do método e a criação em si de um OA completo e robusto que atenda os fins educacionais das disciplinas de Computação e consequentemente do curso como um todo.

Após a criação do OA para PBL, executaremos duas ou três iterações do OA criado no contexto do ensino de Computação nos cursos de engenharia do Campus Campo Mourão da UTFPR. Esta quantidade de iterações, dependerá da disponibilidade de tempo durante o primeiro e ou segundo semestres do ano letivo de 2016.

Ainda vamos avaliar o esforço para desenvolvimento do OA, considerando a abordagem LOD em suas fases: modelagem conceitual, modelagem instrucional e modelagem de interação. Desta última modelagem é que os requisitos do OA serão conhecidos o ai sim este objeto ou objetos serão criados.

Por fim, para avaliar a qualidade dos objetos de aprendizagem (tanto do ponto de vista de professores quanto de alunos), os resultados das três iterações serão analisados e compilados para que, possivelmente, sejam retirados dados e informações de possam realmente constatar os benefícios do uso destes OA na formação destes alunos.

4.3. Cronograma

Visando atingir os objetivos propostos descreve-se as atividades e o cronograma está ilustrados na Tabela 4.1.

1. Identificar elementos de PBL.
   Resultados esperados: Relatório de características de elementos necessários à PBL.
2. Adicionar mecanismos ao LODM para a representação de elementos PBL.
   Resultados esperados: Esteriótipo PBL e manutenção do LODM.
3. Definir ou revisar protocolo para realização de estudo de caso sobre desenvolvimento e utilização de objeto de aprendizagem.
   Resultados esperados: Protocolo para execução do estudo de caso

4. Desenvolver objeto de aprendizagem com elementos PBL para o ensino de Programação para Engenharia.

   Resultados esperados: Objeto de aprendizagem desenvolvido com o LODM.

5. Avaliar o desenvolvimento e utilização do objeto de aprendizagem.

   Resultados esperados: Avaliação da qualidade do objeto de aprendizagem pela perspectiva do aluno e professor.

6. Disponibilizar objeto de aprendizagem em repositório de objetos de aprendizagem ou de recursos educacionais abertos.

   Resultados esperados: Objeto de aprendizagem disponível publicamente, artigo sobre o objeto de aprendizagem.

7. Consolidar LODM para PBL e sintetizar resultado dos estudos de casos realizados.

   Resultados esperados: Artigo sobre LODM para PBL.

8. Redação de dissertação.

   Resultados esperados: Dissertação de mestrado

9. Defesa de dissertação.

   Resultados esperados: Apresentação e defesa de dissertação de mestrado.

Tabela 4.1: Cronograma das atividades

Ano	2016											2017	
Mês / Atividade	Fev	Mar	Abr	Mai	Jun	Jul	Ago	Set	Out	Nov	Dez	Jan	Fev
1													
2													
3													
4													
5													
6													
7													
8													
9													

Referências

AUDINO, Daniel Fagundes. *Objetos de aprendizagem hipermídia aplicado à cartografia escolar no sexto ano do ensino fundamental em geografia*. 152 p. Dissertação (Mestrado) — Universidade Federal de Santa Catarina, Florianópolis, SC,Brasil, jan. 2012. Disponível em: `https://repositorio.ufsc.br/bitstream/handle/123456789/99501/303100.pdf?sequence=1`.

AURELIANO, Viviane Cristina Oliveira; TEDESCO, Patrícia Cabral de Azevedo Restelli. Ensino-aprendizagem de programação para iniciantes: uma revisão sistemática da literatura focada no SBIE e WIE. In: *23º Simpósio Brasileiro de Informática na Educação*. Rio de Janeiro, RJ,Brasil: SBC, 2012. p. 1–10. ISSN 2316-6533.

AVOURIS, N.; KAXIRAS, S.; KOUFOPAVLOU, O.; SGARBAS, K.; STATHOPOULOU, P. Teaching introduction to computing through a project-based collaborative learning approach. In: *14th Panhellenic Conference on Informatics*. Tripoli, Líbia: IEEE, 2010. p. 237–241. ISBN 978-1-4244-7838-5.

BARBOSA, Ellen Francine. *Uma Contribuição ao Processo de Desenvolvimento e Modelagem de Módulos Educacionais*. 253 p. Tese (Doutorado) — Instituto de Ciências Matemáticas e de Computação – Universidade de São Paulo, São Carlos, SP,Brasil, mar. 2004.

BARBOSA, E. F.; MALDONADO, José Carlos. An integrated content modeling approach for educational modules. In: KUMAR, Deepak; TURNER, Joe (Ed.). *IFIP International Conference on Education for the 21st Century - 19th IFIP World Computer Congress*. Santiago,: Springer, 2006. v. 210, p. 17–26.

BARBOSA, E. F.; MALDONADO, J. C. Towards the establishment of a standard process for developing educational modules. In: *36th Annual Frontiers in Education Conference*. San Diego, CA, EUA: IEEE, 2006. p. 5–10. ISBN 1-4244-0256-5. ISSN 0190-5848.

BARBOSA, Ellen Francine; MALDONADO, José Carlos. IMA-CID: an integrated modeling approach for developing educational modules. *Journal of the Brazilian Computer Society*, Springer, Heidelberg,Alemanha, v. 17, n. 4, p. 207–239, nov. 2011. ISSN 0104-6500, 1678-4804.

BARBOSA, E. F.; MALDONADO, J. C.; MAIDANTCHIK, C. L. L. Padronização de processos para o desenvolvimento de módulos educacionais. In: *XXIX Latin-American Conference on Informatics*. La Paz,: [s.n.], 2003. p. 1–12.

BEHRENS, A.; ATORF, L.; SCHWANN, R.; NEUMANN, B.; SCHNITZLER, R.; BALLè, J.; HEROLD, T.; TELLE, A.; NOLL, T. G.; HAMEYER, K.; AACH, T. MATLAB meets LEGO mindstorms - a freshman introduction course into practical engineering. *Transactions on Education*, IEEE Education Society, Los Alamitos, CA,EUA, v. 53, n. 2, p. 306–317, maio 2010. ISSN 0018-9359.

BERBEL, Neusi Aparecida Navas. A problematização e a aprendizagem baseada em problemas: diferentes termos ou diferentes caminhos? *Interface – Comunicação, Saúde, Educação*, UNESP, Botucatu, SP,Brasil, v. 2, n. 2, p. 139–154, fev. 1998. ISSN 1807-5762.

BIOLCHINI, Jorge; MIAN, Paula Gomes; NATALI, Ana Candida Cruz; TRAVASSOS, Guilherme Horta. *Systematic Review in Software Engineering*. Rio de Janeiro, RJ,Brasil, maio 2005. Disponível em: `http://alarcos.inf-cr.uclm.es/doc/MetoTecInfInf/Articulos/es67905.pdf`.

BOND, Stephen T.; INGRAM, Caroline; RYAN, Steve. Reuse, repurposing and learning design - lessons from the DART project. v. 50, n. 2, p. 601–612, fev. 2008.

DEELMAN, A.; HOEBERIGS, B. A ABP no contexto da Universidade de Maastricht. In: SUMMUS (Ed.). *Aprendizagem Baseada em Problemas no Ensino Superior*. 1. ed. São Paulo, SP,Brasil: In: ARAÚJO, Ulisses F; SASTRE, Genoveva (Orgs.). Aprendizagem Baseada em Problemas: no ensino superior, 2009. p. 79–100. ISBN 978-85-323-0532-9.

DYNE, M. van; BRAUN, J. Effectiveness of a computational thinking (CS0) course on student analytical skills. In: *45th ACM technical symposium on Computer science education*. New York, NY,EUA: ACM, 2014. p. 133–137. ISBN 978-1-4503-2605-6.

EDGCOMB, Alex Daniel; VAHID, Frank. Effectiveness of online textbooks vs. interactive web-native content. In: *121st ASEE Annual Conference and Exposition*. Indianapolis, IN,EUA: American Society for Engineering Education, 2014. p. 1–9.

FAN, K. Y. D.; SCHWARTZ, D. I. First programming course in engineering: Balancing tradition and application. In: *ASEE 2003 Annual Conference*. Nashville, Tennessee,EUA: ASEE, 2003. p. 55–60. ISSN 0190-1052.

FREIRE, Paulo Reglus Neves. *Pedagogia da autonomia: saberes necessárias à prática educativa*. 39. ed. São Paulo, SP,Brasil: Paz e Terra, 1996. 148 p. (Coleção Leitura). ISBN 978-85-7753-015-1.

GIL, Antonio Carlos. *Métodos e técnicas de pesquisa social*. 6. ed. São Paulo, SP,Brasil: Atlas, 2008. 200 p. ISBN 978-85-224-5142-5.

GIL, Antonio Carlos. *Didática do Ensino Superior*. São Paulo, SP,Brasil: Atlas, 2015. 304 p. São Paulo. ISBN 978-8522443925.

KICK, Richard; TREES, Frances P. AP CS Principles: Engaging, challenging, and rewarding. *Inroads*, ACM, New York, NY, EUA, v. 6, n. 1, p. 42–45, fev. 2015. ISSN 2153-2184, 2153-2192.

KITCHENHAM, Barbara; CHARTERS, Stuart. *Guidelines for performing Systematic Literature Reviews in Software Engineering*. Reino Unido, jul. 2007. 65 p.

KITCHENHAM, Barbara A.; DYBA, Tore; JORGENSEN, Magne. Evidence-based software engineering. In: *26th Internacional Conference on Software Engineering*. [S.l.: s.n.], 2004. p. 273–281. ISBN 0-7695-2163-0. ISSN 0270-5257.

KROLL, Per; KRUCHTEN, Philippe. *The Rational Unified Process Made Easy: A Practitioner's Guide to the RUP*. 1. ed. Boston, MA,EUA: Addison-Wesley Longman, 2003. 464 p. (Object Technology Series). ISBN 0-321-16609-4.

LIANG, H. N.; FLEMING, C.; MAN, K. L.; TILLO, T. A first introduction to programming for first-year students at a Chinese university using LEGO MindStorms. In: *2nd International Conference on Teaching, Assessment and Learning for Engineering*. Bali,|ndonésia: IEEE, 2013. p. 233–238. ISBN 9781467363556.

MAHADEVAN-JANSEN, A.; ROWE, C. J.; CROCETTI, J.; BROPHY, S. A paradigm shift in the approach to freshman engineering education. In: *ASEE 2003 Annual Conference*. Nashville, Tennessee,EUA: ASEE, 2003. p. 11570–11585. ISSN 0190-1052.

MALAN, D. J. Reinventing CS50. In: *41st ACM Technical Symposium on Computer Science Education*. Milwaukee, WI,EUA: ACM, 2010. p. 152–156. ISBN 9781605588858.

MAMEDE, Silvia. Aprendizagem baseada em problemas: Características, processos e racionalidade. In: *Aprendizagem baseada em problemas: anatomia de uma abordagem educacional*. Fortaleza, CE,Brasil: Hucitec, 2001. cap. 2, p. 27–48. ISBN 9788527105712.

NOOR, N. M.; HARUN, J.; ARIS, B. Application of the pedagogical and andragogical model in web-based learning instruction among non-major computer science students' learning programming. In: *2014 International Conference on Teaching and Learning in Computing and Engineering*. Kuching, Sarawak.: IEEE Computer Society, 2014. p. 106–111. ISBN 9781479935918.

NOVAK, Joseph Donald. *A theory of education*. Ithaca, NY,EUA: Cornell University Press, 1977. 295 p. ISBN 9780801411045.

RECKINGER, S. M.; RECKINGER, S. J. An interactive programming course model for mechanical engineering students. In: *121st ASEE Annual Conference and Exposition*. Indianapolis, IN,EUA: American Society for Engineering Education, 2014. p. 1–22.

RIBEIRO, Luis Roberto de Camargo. *A aprendizagem baseada em problemas (PBL): uma implementação na educação em Engenharia na voz dos atores*. 236 p. Tese (Doutorado) — UFSCar, São Carlos, SP, Brasil, 2005.

RIBEIRO, Luis R. de Camargo. *Aprendizagem baseada em problemas (PBL): uma experiência no ensino superior*. 1. ed. São Carlos, SP,Brasil: EdUFSCar, 2008. 151 p. ISBN 9788576001140.

SARRIA, M.; GERALDO, M. Introduction to programming for engineers following the parachute paradigm. In: *39th IEEE Frontiers in Education Conference*. San Antonio, TX,EUA: IEEE, 2009. p. 1–4. ISSN 0190-5848.

SCHMITZ, Carsten *et al. LimeSurvey*. maio 2007. Programa de computador. Disponível em: `https://www.limesurvey.org/`.

SILVA, Marco Aurélio Graciotto. *LOD: uma abordagem para desenvolvimento de objetos de aprendizagem multimídias e interativos*. Tese (Doutorado) — Universidade de São Paulo, São Carlos, SP,Brasil, mar. 2012.

SILVA, Marco Aurélio Graciotto; BARBOSA, Ellen Francine; MALDONADO, José Carlos. Model-driven development of learning objects. In: *41st ASEE/IEEE Frontiers in Education Conference*. Rapid City, South Dakota, EUA: IEEE, 2011. (FIE), p. F4E–1 – F4E6. ISBN 978-1-61284-467-1. ISSN 0190-5848.

SPINELLI, Walter. *Aprendizagem Matemática em Contextos Significativos: Objetos Virtuais de Aprendizagem e Percursos Temáticos*. 123 p. Dissertação (Mestrado) — Faculdade de Educação – Universidade de São Paulo, São Paulo, SP,Brasil, nov. 2005.

SUN, L.; KINDY, M.; LIRON, C.; GRANT, C. D.; WATERHOUSE, S. A. Hybrid course design: Leading a new direction in learning programming languages. In: *119th ASEE Annual Conference and Exposition*. San Antonio, TX,EUA: ASEE, 2012. p. 1–13. ISBN 9780878232413.

WALKER, Henry M. Priorities for the non-majors, CS course: Programming may not make the cut. *Inroads*, ACM, New York, NY, EUA, v. 6, n. 1, p. 46–49, fev. 2015. ISSN 2153-2184, 2153-2192.

WILEY, David A. Connecting learning objects to instructional design theory: A definition, a metaphor, and a taxonomy. In: WILEY, David A. (Ed.). *The Instructional Use of Learning Objects*. 1. ed. EUA: AIT/AECT, 2000. cap. 1.1, p. 1–35. ISBN 978-0784208922. Disponível em: `http://reusability.org/read/chapters/wiley.doc`.

WILEY, David A. *Learning Object Design and Sequencing Theory*. 131 p. Tese (Doutorado) — Department of Instructional Psychology and Technology – Brigham Young University, Provo, Utah,EUA, jun. 2000. Disponível em: `http://opencontent.org/docs/dissertation.pdf`.

YIN, Robert K. *Estudo de caso: planejamento e métodos*. 2. ed. Porto Alegre, RS,Brasil: Bookman, 2001. 205 p. ISBN 85-7307-852-9.

ZACHARY, Joseph L. Introduction to computing for engineers: New approaches to content and pedagogy. In: *26th Annual Conference on Frontiers in Education*. Piscataway, NJ,EUA: IEEE, 1996. p. 149–153. ISBN 0-7803-3348-9. ISSN 0190-5848.

Apêndices

Questionários

Foram disponibilizados questionários para alunos (12 questões), coordenadores (7 questões), professores do Departamento de Computação (8 questões) e professores de outro Departamentos (6 questões). Foi utilizada a ferramenta LimeSurvey (SCHMITZ *et al.*, 2007), um software livre desenvolvido com o objetivo de preparar, publicar e coletar respostas de questionários.

Nestas questões, alunos, coordenadores e professores, puderam expor suas visões em relação à disciplina de Computação em cada um de seus cursos, subsidiando os resultados e conclusões apresentados na Seção 2.2.

Nas páginas a seguir são apresentados os questionários aplicados neste estudo.

APÊNDICE I

Questionário ALUNOS

Prezado Aluno / Aluna, obrigado por colaborar com esta pesquisa. Completar este breve questionário vai nos ajudar a obter os melhores resultados em relação ao **Levantamento sobre a situação das disciplinas introdutórias à computação em cursos de Engenharia.**

1 - Em qual Curso esta matriculado?
( ) Engenharia Ambiental
( ) Engenharia de Alimentos
( ) Engenharia Civil

2 - Qual disciplina você cursou?
( ) Computação I
( ) Fundamentos de Programação
( ) Não fiz disciplina de Programação
( ) Já havia feito em outra instituição (Convalidação). Informe a(s) disciplina (s):
_____

3 - Você já cursou a disciplina de CÁLCULO NUMÉRICO?
( ) Sim
( ) Não

4 - Você já cursou a disciplina de GEOPROCESSAMENTO?
( ) Sim
( ) Não

5 - Você já cursou a disciplina de TEORIA DAS ESTRUTURAS 2?
( ) Sim
( ) Não

6 - Qual a importância da disciplina de Programação em relação às demais disciplinas do seu curso?
( ) Extremamente importante (essencial)
( ) Importante
( ) Nada importante

7 - Caso tenha alguma importância, qual foi? Descreva.
_____
_____
_____

8 - Ordene os seguintes conceitos de acordo com a importância para sua formação de engenheiro.

Clique duas vezes ou arraste os itens na coluna esquerda para move-los para a direita, ordenando, de cima para baixo, da prioridade mais alta para a mais baixa.

Suas opções

- Criatividade (Computação é uma atividade criativa)
- Abrastração (Abstração reduz informações e detalhes para facilitar o foco em conceitos relevantes)
- Dados e Informações (Dados e informações facilitam a criação de conhecimento)
- Algoritmos (O desenvolvimento, uso e análise de algoritmos são alguns dos aspectos mais fundamentais da computação)
- Programação (Programação permite a resolução de problemas, a expressão humana, e criação de conhecimento)
- A Internet (A Internet permeia computação moderna. Os sistemas construídos sobre ela tem um profundo impacto na sociedade)
- Impacto Global (Nossos métodos de comunicação, colaboração, resolução de problemas, e fazer negócios mudaram e estão mudando devido às inovações possibilitadas pela computação)

Sua classificação

9 - Caso tenha algum comentário adicional sobre a importância dos conceitos de Computação, descreva-o.

_____
_____
_____

10 - Você já participou de Projeto de Pesquisa, Projeto de Extensão ou Estágio?

( ) Projeto de Pesquisa
( ) Projeto de Extensão
( ) Estágio
( ) Nenhuma das opções

11 - Qual foi a importância da disciplina de Programação em relação aos projetos de Pesquisa, Extensão ou Estágio?

( ) Extremamente importante (essencial)
( ) Importante
( ) Nada importante
( ) Não se aplica (sem atividade de estágio, iniciação científica ou extensão)

12 - Caso tenha alguma importância, qual foi? Descreva.

_____
_____
_____

APÊNDICE II

Questionário COORDENADORES

Prezado Sr. / Sra., obrigado por colaborar com esta pesquisa. Completar este breve questionário vai nos ajudar a obter os melhores resultados em relação ao **Levantamento sobre a situação das disciplinas introdutórias à computação em cursos de Engenharia.**

1 - Qual curso coordena ou coordenou?
( ) Engenharia Ambiental
( ) Engenharia de Alimentos
( ) Engenharia Civil

2 - Qual a importância da disciplina de Computação no seu curso?
( ) Extremamente importante (essencial)
( ) Importante
( ) Nada importante

3 - Caso tenha alguma importância, qual foi? Descreva.
_____
_____
_____

4 - Em relação aos pré-requisitos, além daqueles explícitos na grade, existem outros quanto à disciplina de Computação? Quais?
_____
_____
_____

5 - Você tem informação se a disciplina de Computação é EXIGIDA pela legislação da UTFPR para o seu curso?
( ) Não
( ) Sim
Comente:
_____
_____
_____

6 - Você tem informação se a disciplina de Computação é EXIGIDA pela legislação Externa(CNE, Catálogo de Cursos, Conselhos Profissionais, etc.) para o seu curso?
( ) Não
( ) Sim

Comente:

_____
_____
_____

7 - Sugestões ou comentários:

_____
_____
_____

APÊNDICE III

Questionário PROFESSORES do DACOM

Prezado Sr. / Sra., obrigado por colaborar com esta pesquisa. Completar este breve questionário vai nos ajudar a obter os melhores resultados em relação ao **Levantamento sobre a situação das disciplinas introdutórias à computação em cursos de Engenharia.**

1 - Quantas vezes você já ministrou a disciplina de Computação I ou Fundamentos de Programação?
( ) UMA vez
( ) DUAS vezes
( ) TRÊS vezes
( ) QUATRP vezes
( ) MAIS que quatro

2 - Para que cursos você já ministrou tais disciplinas?
( ) Engenharia Ambiental
( ) Engenharia de Alimentos
( ) Engenharia Civil
( ) Outros: _____

3 - Ordene as principais dificuldades encontradas.

Clique duas vezes ou arraste os itens na coluna esquerda para move-los para a direita, ordenando, de cima para baixo, da prioridade mais alta para a mais baixa.

Suas opções

- Programa
- Abordagem
- Linguagem de programação utilizada
- Comportamento e interesse dos alunos
- Turmas misturadas
- Alunos em dependência
- Mecanismos de avaliação

Sua classificação

4 - Fique a vontade para justificar a pergunta anterior, descrever outros dificuldades e fazer comentários.

_____
_____
_____

5 - Ordene os Instrumentos e Técnicas utilizados em aula de acordo com sua importância.

Suas opções

- Quadro branco
- Projetor
- Simulação e visualização de algoritmos
- Exemplos
- TDD (Desenvolvimento baseado em testes)
- Competições de programação
- Aprendizagem baseada em problemas
- Programação em pares

Sua classificação

6 - Fique a vontade para justificar a pergunta anterior, descrever outros instrumentos e técnicas utilizados em aula e fazer comentários.

_____
_____
_____

7 - Qual é a importância da disciplina de Programação para os cursos de Engenharia Ambiental, Engenharia de Alimentos e Engenharia Civil?

( ) Extremamente importante (essencial)
( ) Importante
( ) Nada importante

8 - Caso tenha alguma importância, qual foi? Descreva.

_____
_____
_____

APÊNDICE IV

Questionário PROFESSORES outros Departamentos

Prezado Sr. / Sra., obrigado por colaborar com esta pesquisa. Completar este breve questionário vai nos ajudar a obter os melhores resultados em relação ao **Levantamento sobre a situação das disciplinas introdutórias à computação em cursos de Engenharia.**

1 – Em qual Departamento esta lotado?

Por favor, selecione:
DAAMB
DACOC
DAELN
DAFIS
DAGEE
DAHUM
DALIM
DAMAT
DAQUI

2 - Você ministrou alguma disciplina que dependia de conhecimento e habilidade de Computação/Programação?

( ) NÃO
( ) SIM
( ) Se Sim, quais: _____
_____

3 - Quais são as principais dificuldades encontradas em relação a Computação nessas disciplinas?

_____
_____
_____

4 - Quais os pontos positivos ou destaques em relação aos conhecimentos e habilidades dos alunos quanto à Computação nessas disciplinas?

_____
_____
_____

5 - Em relação à projetos (pesquisa, extensão), as dificuldades foram as mesmas relatadas quanto às disciplinas? Caso não sejam, descreva-as.

( ) Sim
( ) Não
( ) Quais foram :_____
_____

6 - Sugestões ou comentários:

_____
_____
_____