A Practical and Systemic Curricular Approach to Teach Computer Systems

T. Oliveira, D. Stringhini, and J. Craibas

***Abstract*—The design of computational systems is an important matter that must be especially present in Computer Engineering curriculum. Traditionally, Computer Engineering curriculum address this theme in a compartmentalized way in specific curricular units. This organization creates a fragmented view of the complex computational systems development, understood here as the specification, design, implementation and integration of the lines of computer architecture and organization, systems engineering, compilers, operating systems and computer networks. To avoid this fragmented view, this article proposes a systemic and curricular approach, integrating hardware and software, which allows students to design a complex computer system over three years of their academic trajectories. This approach was validated in a Computer Engineering course at the Federal University of São Paulo, bringing positive results in relation to students' motivation and enthusiasm, as well as encouraging creative and innovative thinking.**

***Index Terms*—Teaching of computer systems, Curricular and systemic approach, Hardware and software designs.**

I. Introdução

UM engenheiro deve ter a capacidade de especificar, desenvolver e implementar sistemas. Neste sentido, a sociedade brasileira de computação (SBC) e institutos internacionais, como ACM (*Association for Computing Machinery*) e IEEE (*Institute of Electrical and Electronics Engineers*), têm buscado publicar referenciais orientadores de formação para cursos de graduação em Computação [1] [2], especificando a importância de um engenheiro de computação em projetar, desenvolver e implementar sistemas computacionais. O aluno de engenharia de computação deve ser capaz de construir hardware, software, sistemas de comunicações e suas interações, seguindo teorias, princípios, métodos, técnicas e procedimentos da engenharia e da computação. Além disso, o aluno deve adquirir competências e habilidades que lhe permitam realizar estudos, planejar, especificar, projetar, desenvolver e implementar sistemas computacionais de propósito geral ou específico.

Dentro deste contexto, muitos currículos de Engenharia de Computação apresentam em sua matriz curricular algumas unidades curriculares (disciplinas, matérias ou cadeiras) relacionadas ao desenvolvimento de hardware e de software. Apesar de não haver um padrão para os nomes das unidades curriculares nos diversos cursos de Engenharia de Computação, os conteúdos abordados envolvem: Circuitos Digitais, Arquitetura e Organização de Computadores, Linguagens Formais e Autômatos, Compiladores, Sistemas Operacionais, Redes de Computadores, entre outros.

No entanto, apesar desses currículos abordarem o tema relacionado ao projeto e implementação de sistemas computacionais, as unidades curriculares previstas em seus respectivos projetos pedagógicos não costumam ser interligadas, e assim os alunos acabam adquirindo uma visão fragmentada de um sistema computacional complexo.

Entenda-se aqui, neste artigo, como sistema computacional complexo o envolvimento e a interligação de todos os conteúdos curriculares comentados nos parágrafos anteriores (as linhas de arquitetura e organização de computadores, engenharia de sistemas, compiladores, sistemas operacionais e redes de computadores) e suas correlações, abrangendo a integração de aspectos tanto de hardware quanto de software.

Para evitar que os alunos tenham essa visão fragmentada no desenvolvimento de um sistema computacional que envolva tanto hardware quanto software, no currículo do curso de Engenharia de Computação da Universidade Federal de São Paulo (Unifesp) [3] as unidades curriculares são subdivididas em dois grandes grupos: as unidades curriculares gerais e as unidades curriculares integradas.

Tanto as unidades curriculares gerais quanto as unidades curriculares integradas possuem conteúdos (ementas) que estão relacionados à formação técnica do engenheiro de computação, permitindo o desenvolvimento de competências e habilidades definidas no perfil do aluno egresso. No entanto, as unidades curriculares integradas possuem uma função pedagógica e didática fundamental e inovadora para a formação de um profissional diferenciado e bem qualificado. As unidades curriculares integradas são utilizadas para que o aluno possa, de fato, desenvolver um sistema computacional complexo durante o seu processo de aprendizagem no decorrer do curso. A cada semestre o aluno deverá desenvolver uma parte do sistema computacional, finalizando-o após seis semestres ou três anos de curso.

Sendo assim, tendo em vista a ocorrência dessa visão fragmentada nos currículos de curso de Engenharia de Computação, o objetivo deste artigo é propor uma estrutura curricular diferenciada, capaz de permitir que os alunos desenvolvam, ao longo de suas trajetórias acadêmicas, um sistema computacional complexo.

Na sequência, na seção II, encontram-se uma compilação dos trabalhos relacionados e das práticas pedagógicas em-

T. Oliveira está lotado na Universidade Federal de São Paulo, Instituto de Ciência e Tecnologia, São José dos Campos, São Paulo, Brasil (e-mail: tiagoooli@yahoo.com.br).

D. Stringhini está lotada na Universidade Federal de São Paulo, Instituto de Ciência e Tecnologia, São José dos Campos, São Paulo, Brasil (e-mail: dstring@gmail.com).

J. J. S. Craibas está matriculado na Universidade Federal de São Paulo, Instituto de Ciência e Tecnologia, São José dos Campos, São Paulo, Brasil (e-mail: jailson.craibas@gmail.com).

pregadas no ensino das unidades curriculares comentadas anteriormente. Na seção III, apresentam-se as unidades curriculares integradas elaboradas para o curso de Engenharia de Computação da Unifesp, bem como a sua decorrente metodologia de ensino-aprendizagem. Na seção IV, discorre-se sobre a qualidade educacional que vem sendo obtida com a aplicação dessa estrutura curricular e, por fim, na seção V, encontram-se as considerações finais.

II. Trabalhos Relacionados

Diversas práticas educacionais para a realização de projetos pelos alunos têm sido propostas na literatura científica relacionadas ao ensinamento de conceitos que envolvem sistemas computacionais.

Na linha de arquitetura e organização de computadores, podem-se citar os trabalhos publicados em [4], [5] e [6]. Em Kellett [4], o autor descreve a metodologia utilizada em uma disciplina onde um determinado projeto relacionado à arquitetura e organização de computadores é inicialmente especificado, devendo ser desenvolvido pelos alunos durante a realização da disciplina. Em Cifredo-Chacõn *et al.* [5], os autores propõem o uso de metodologia ativa (*learning-by-doing*) para o desenvolvimento e implementação em FPGA (*Field Programmable Gate Array*) de um processador, onde conceitos de dispositivos lógicos programáveis e da linguagem de descrição de hardware VHDL (*Very High Speed Integrated Circuits Hardware Description Language*) são introduzidos aos alunos. Em Larraza-Mendiluze *et al.* [6] [7], os autores descrevem o uso do console de videogame Nintendo em um ambiente de aprendizagem baseado em problema ou projeto (*Problem-based learning* - PBL ou *Project-based learning* - PjBL), onde a partir de um projeto base referente ao desenvolvimento de um determinado jogo, os alunos são estimulados a aprender conceitos de subsistemas de Entrada/Saída (E/S) como, por exemplo, E/S programada (*polling*) ou E/S orientada à interrupção.

Na linha de engenharia de sistemas, podem-se citar os trabalhos publicados em [8] e [9]. Em Billard [8], o autor propõe o projeto e a implementação de um sistema operacional didático específico por meio da linguagem UML (*Unified Modeling Language*), metodologia orientada a objetos e linguagem JAVA. Por sua vez, no trabalho descrito em Ortin *et al.* [9], os autores utilizam padrões de projeto utilizando a linguagem UML numa disciplina de construção de um compilador.

Na linha de compiladores, Baldwin [10] traz uma linguagem de programação simplificada e seu respectivo compilador com características modulares, facilitando o seu entendimento e projeto numa disciplina de graduação. Em Mernik e Zumer [11], apresenta-se uma ferramenta didática denominada LISA para facilitar o aprendizado dos conceitos envolvidos na construção e projeto de compiladores. Por sua vez, em Kundra e Sureka [12][13], apresenta-se uma metodologia de ensino-aprendizagem (*case-based teaching*) aplicado a uma disciplina de projeto de compiladores.

Na linha de sistemas operacionais, podem-se citar os trabalhos publicados em [14], [15] e [16]. Em Garmpis [14], uma ferramenta didática via web foi descrita para auxiliar os alunos no entendimento sobre algoritmos de substituição de páginas. Por sua vez, em Jong *et al.* [15], os autores trabalham com a aplicação de jogos em uma disciplina de sistemas operacionais para melhorar a motivação dos alunos quanto ao conteúdo programático apresentado. Em Wong [16], uma metodologia de ensino-aprendizagem ativa (*maker movement education*) é utilizada para o ensino de sistemas operacionais, onde os alunos devem trabalhar com um robô segue-faixa implementado por meio de um microcontrolador Raspberry Pi.

Por fim, na linha de redes de computadores, em El-Kharashi *et al.* [17], os autores propõem práticas de ensino que envolvem a implementação de uma camada de enlace de dados numa disciplina de projeto em redes de computadores. Em Sarkar [18], diversos trabalhos para serem realizados em laboratório são propostos para o aprendizado de conceitos e fundamentos relacionados a redes de computadores. No trabalho publicado em Winarno [19], os autores utilizam a aprendizagem multimídia (*multimedia learning*) em conjunto com metodologia baseada em problemas (*problem-based learning*) numa disciplina de redes de computadores.

Buscando uma integração maior entre algumas dessas linhas, no trabalho publicado em Abe *et al.* [20], os autores propõem uma disciplina de laboratório oferecida no segundo semestre do primeiro ano de um curso de Ciência da Computação envolvendo a realização de um sistema computacional em FPGA que inclui a organização de um processador, o desenvolvimento de um compilador e a execução de uma aplicação de comunicação em redes. Em linhas gerais, o projeto previamente elaborado e desenvolvido pelos autores é disponibilizado aos alunos sem alguns módulos ou contendo módulos que precisam ser modificados ou adaptados. Os alunos são divididos em grupos, onde cada grupo se responsabiliza pela realização de uma parte do sistema (ou o processador ou o compilador ou a aplicação de comunicação em rede). A cada aula, os alunos são conduzidos por meio da realização de experimentos ou tutoriais a ir construindo os módulos que estão faltando ou a ir modificando ou adaptando os módulos necessários para que, no final da disciplina, o sistema computacional possa ser completamente integrado e a aplicação de comunicação em rede possa ser corretamente executada.

Indo em direção a uma integração no nível curricular, encontram-se os trabalhos publicados em [21], [22], [23] e [24].

Em Santos e Silva [21], os autores abordam a implantação de módulos integradores (denominados MIs) em um curso de Engenharia de Computação. Os MIs fazem uso do método PBL e são realizados em paralelo com o método tradicional de aulas expositivas. Na matriz curricular do curso, os alunos trabalham na resolução de vários problemas/projetos distribuídos em nove MIs durante a trajetória acadêmica dos alunos referentes à programação, engenharia de software, projeto de circuitos digitais, conectividade e concorrência, entre outros assuntos. Embora ocorra uma maior padronização e uniformidade em relação à metodologia de ensino-aprendizagem aplicada, a fragmentação comentada na seção I ainda ocorre,

já que não há uma integração entre os MIs realizados a cada semestre.

Por sua vez, o currículo de curso de engenharia elétrica e de computação proposto em Somerville *et al.* [22] e disponível na página oficial da Universidade [25] apresenta várias inovações, dentre as quais, os alunos possuem disciplinas integradas com duas ou mais áreas de conhecimento, trabalham na resolução de problemas reais e realizam estudos nas áreas de artes, humanidades e ciências sociais. Semelhante ao que ocorre com os MIs no trabalho publicado em Santos e Silva [21], os alunos devem se matricular em disciplinas específicas com o intuito de desenvolver vários projetos durante o seu percurso acadêmico referentes à arquitetura e organização de computadores, desenvolvimento de softwares, processamento de sinais, comunicação digital e analógica, entre outros assuntos.

Em Rehman *et al.* [23], os autores propõem uma estrutura curricular onde os projetos realizados pelos alunos são classificados da seguinte forma: os instrutores conduzem passo-a-passo os alunos na realização do projeto (*structured design experience* - SD); os instrutores procuram apenas guiar os alunos, sem, no entanto, disponibilizar o procedimento exato para a realização do projeto (*guided design experience* - GD) e; os alunos possuem uma maior liberdade na definição, especificação e realização do projeto (*open-ended design experience* - OE). No artigo, define-se que, no começo do curso, os alunos devam ser expostos a projetos do tipo SD, passando a realizar projetos do tipo GD e, por fim, devam terminar o curso realizando projetos do tipo OE.

De maneira semelhante, em Rashid e Tasadduq [24], os autores trazem uma proposta de currículo baseada na metodologia *Y-chart*, dividindo, basicamente, os conteúdos curriculares de um curso de Engenharia de Computação em dois grandes níveis, denominados componente e sistema. No nível de componente, os conteúdos curriculares são subdivididos em dois grupos isolados: um relacionado a hardware e o outro relacionado a software. Por sua vez, no nível de sistema busca-se realizar a integração entre o nível de componente de hardware e o nível de componente de software. Além disso, as aulas de laboratório propostas ao longo do percurso acadêmico dos alunos são realizadas de três maneiras distintas, sendo: completamente baseadas em procedimentos contendo o passo-a-passo para a resolução do projeto (*procedure-based design experience*); contendo práticas realizadas pelos alunos de forma semi-independente (*semi-independent design experience*) e; completamente independentes (*completely independence design experience*).

As propostas curriculares apresentadas nesses trabalhos trazem diversas inovações pedagógicas, seja na metodologia de ensino-aprendizagem empregada [21], seja na integração entre duas ou mais áreas de conhecimento [22], ou seja na estruturação de disciplinas práticas de laboratório [23] [24]. Dentro desse contexto, a proposta apresentada neste artigo busca uma integração ainda maior, prevendo em sua estrutura curricular a reutilização de projetos desenvolvidos anteriormente em disciplinas de laboratório, onde o aluno terá a possibilidade de construir do zero, em três anos durante seu percurso acadêmico, e com alto grau de liberdade, um sistema computacional complexo, associando e correlacionando as linhas de arquitetura e organização de computadores, engenharia de sistemas, compiladores, sistemas operacionais e redes de computadores.

Resumidamente, na Tabela I, encontra-se uma compilação das principais características das práticas educacionais empregadas nos trabalhos relacionados comentados anteriormente.

TABELA I
CARACTERÍSTICAS DAS PRÁTICAS EDUCACIONAIS APLICADAS NA REALIZAÇÃO DE PROJETOS RELACIONADOS AO ENSINAMENTO DE CONCEITOS QUE ENVOLVEM SISTEMAS COMPUTACIONAIS

Abordagens e Estratégias Adotadas	**Multidisciplinaridade**	**Integração Curricular**	**Transversalidade de Projeto**
Kellett [4], Baldwin [10] e El-Kharashi *et al.* [17]	Não	Não	Não
Larraza-Mendiluze *et al.* [7], Billard [8] e Wong [16]	Sim	Não	Não
Abe *et al.* [20]	Dependente da interação alcançada entre os grupos de alunos formados	Não	Não
Santos e Silva [21]	Dependente do problema PBL aplicado	Sim, por meio dos módulos integradores (MIs)	Não
Somerville *et al.* [22]	Sim	Sim	Não
Rehman *et al.* [23]	Dependente do projeto a ser estabelecido	Sim, por meio da estruturação entre projetos SD, GD e OE	Não
Rashid e Tasadduq [24]	Sim, no nível de sistema quando da interligação entre os componentes de hardware e de software	Sim, pelo uso da metodologia Y-chart adotada	Não
Este artigo	Sim	Sim	Sim

Na primeira coluna, tem-se alguns exemplos que caracterizam as abordagens e estratégias adotadas. Na segunda coluna, discrimina-se a multidisciplinaridade da proposta. Entenda-se aqui, neste artigo, como multidisciplinaridade a possibilidade de desenvolvimento de um projeto de sistema computacional envolvendo duas ou mais linhas de conhecimento, sendo elas: arquitetura e organização de computadores, engenharia de sistemas, compiladores, sistemas operacionais e redes de computadores.

Na terceira coluna da Tabela I, identifica-se a integração curricular da proposta, a qual ocorre quando da existência de uma estratégia ou abordagem que leve em consideração a estrutura curricular do curso, fazendo-se uso da interligação entre unidades curriculares correlacionadas de forma horizontal (interligação entre unidades curriculares do mesmo termo/semestre do aluno) ou vertical (interligação entre unidades curriculares de termos/semestres diferentes).

Na última coluna, por sua vez, encontra-se a transversalidade de projeto, definida aqui como sendo a possibilidade de se projetar um sistema computacional complexo (ao envolver a especificação, desenvolvimento, implementação e integração de todas as linhas de conhecimento mencionadas anteriormente) de maneira gradual enquanto da realização da trajetória acadêmica do aluno no curso. Desta forma, o projeto será realizado de forma transversal, onde a cada semestre, após a aquisição de novos conhecimentos sobre determinados conceitos de sistemas computacionais, o aluno irá avançando e progredindo no desenvolvimento desse sistema.

Para que essa transversalidade de projeto possa ser atingida, propõe-se, como estratégia pedagógica, uma nova forma de integração curricular com enfoque especificamente em sistemas computacionais. Essa nova forma de integração curricular utiliza-se, como elemento chave, a reutilização de artefatos ou trabalhos que serão produzidos pelo aluno ao longo de vários anos durante sua trajetória acadêmica no curso.

Na sequência, na seção III, encontra-se a descrição da proposta dessa nova forma de integração curricular que permite a transversalidade de projeto e que busca, como consequência, reduzir a visão fragmentada do aluno no desenvolvimento de um sistema computacional.

III. As Unidades Curriculares Integradas

A. Definição dos Laboratórios Integrados

As unidades curriculares integradas do curso de Engenharia de Computação da Unifesp possuem um papel fundamental na formação acadêmica do aluno, viabilizando uma experiência única e enriquecedora no processo de desenvolvimento de projetos, promovendo a integração entre hardware e software.

No currículo do curso doze unidades curriculares são integradas em um único grupo, quais sejam: “Circuitos Digitais”, “Arquitetura e Organização de Computadores”, “Linguagens Formais e Autômatos”, “Compiladores”, “Sistemas Operacionais”, “Redes de Computadores” e seis laboratórios denominados “Laboratórios de Sistemas Computacionais”.

As unidades curriculares denominadas “Circuitos Digitais”, “Arquitetura e Organização de Computadores”, “Linguagens Formais e Autômatos”, “Compiladores”, “Sistemas Operacionais” e “Redes de Computadores” são utilizadas para que aluno adquira a base teórica necessária para o desenvolvimento de um sistema computacional. Por sua vez, nas unidades curriculares denominadas “Laboratórios de Sistemas Computacionais”, o aluno ao longo de três anos irá desenvolver um sistema computacional complexo.

Esse alinhamento entre as unidades curriculares de ”Laboratórios de Sistemas Computacionais” é organizado no currículo por meio de pré-requisitos, indicando que para um aluno cursar um determinado laboratório de sistemas computacionais, ele precisará ser aprovado no laboratório do semestre anterior. Além de ser aprovado no laboratório do semestre anterior, o aluno também precisará ser aprovado na unidade curricular teórica correspondente ao laboratório que cursará. Por exemplo, para se inscrever na unidade curricular de ”Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores”, o aluno precisará ter, como pré-requisitos, a unidade curricular de ”Laboratório de Sistemas Computacionais: Circuitos Digitais” e a unidade curricular teórica de ”Arquitetura e Organização de Computadores”. Para a integralização do curso de Engenharia de Computação, além das unidades curriculares integradas, foco desse artigo, os alunos também precisam realizar simultaneamente outras unidades curriculares mais gerais, visando o aprendizado de outras áreas de conhecimento, como por exemplo, circuitos elétricos, análise de sinais, banco de dados, cálculo, física entre outras. Para mais informações sobre a estrutura curricular do curso de Engenharia de Computação da Unifesp, contendo as unidades curriculares integradas e as gerais, pode-se consultar o seu Projeto Pedagógico[1].

[1]O projeto pedagógico do curso de graduação em Engenharia de Computação da Unifesp foi aprovado em 2015 e atualizado em 2019, podendo ser encontrado no endereço eletrônico http://www.unifesp.br/campus/sjc/o-curso-engcom/projeto-pedagogico-do-curso.html, acessado em 20/08/2019.

Além de permitir o desenvolvimento de um sistema computacional complexo, os laboratórios de sistemas computacionais propiciam o treinamento do aluno no que se refere à apresentação oral de ideias e a redação de textos técnicos e científicos de forma clara, concisa e objetiva.

O sistema computacional que deve ser desenvolvido pelo aluno ao longo de três anos se assemelha ao esquema apresentado na Fig. 1. De acordo com este esquema, o aluno irá inicialmente desenvolver o projeto digital de um processador e dos seus sistemas de memória e de entrada/saída. Uma vez descrito esse sistema de hardware, o aluno utilizará o conjunto de instruções de baixo nível (código de máquina) desenvolvido para realizar o projeto de uma linguagem de programação que possua uma sintaxe de nível mais alto do que o código de máquina do processador.

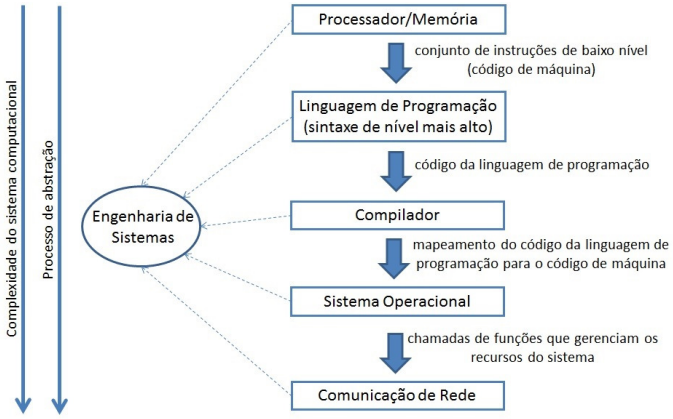

Fig. 1. Diagrama de um sistema computacional integrando o projeto tanto de hardware quanto de software.

Tendo desenvolvido a linguagem de programação do sistema de hardware, o aluno deverá projetar e implementar o sistema de compilação, permitindo que a linguagem de programação possa ser traduzida para seu respectivo código de máquina.

A próxima etapa no desenvolvimento desse sistema computacional é a implementação de um sistema operacional que permita o gerenciamento dos recursos do processador e de seu sistema de memória desenvolvidos nas etapas anteriores, fornecendo, com isso, uma interface entre o sistema de hardware e o usuário.

Por fim, o aluno deverá utilizar as funções disponibilizadas pelo sistema operacional desenvolvido na etapa anterior para a realização de um projeto que envolva a comunicação em rede de dois ou mais sistemas.

Note-se que, de acordo com a Fig. 1, a complexidade e o processo de abstração do sistema computacional crescem ao longo do desenvolvimento do projeto. O aluno inicialmente trabalha no nível de portas lógicas e de circuitos digitais, depois passa a trabalhar num nível mais elevado, quando realiza a etapa de projeto da linguagem de programação e de seu correspondente processo de compilação e, na sequência, começa a trabalhar em um nível ainda mais elevado, quando realiza o projeto de um sistema operacional e de um protocolo para a comunicação em rede.

O desenvolvimento de um sistema computacional deve, sempre, ser pensado como um todo. Os problemas que o sistema deve resolver precisam ser analisados e uma solução envolvendo todos os componentes deve ser proposta. O projeto de cada componente do sistema pode ser conduzido utilizando processos específicos e a engenharia de sistemas deve permear todos esses processos, tornando-se possível a concretização de um projeto de elevada complexidade. Por isso, como mostrado na Fig. 1, a engenharia de sistemas é utilizada durante todas as etapas de projeto. O seu objetivo principal foca na definição das necessidades e funcionalidades do sistema, na realização da documentação sistemática de requisitos e na definição de todo o processo de desenvolvimento, desde a síntese até a validação do sistema, introduzindo-se, para isso, métodos e ferramentas que deverão facilitar a execução do projeto.

Diante do exposto nesta subseção sobre o desenvolvimento do sistema computacional, a abordagem prática e sistêmica de ensino e aprendizado é consolidada efetivamente no currículo do curso de Engenharia de Computação da seguinte forma: no terceiro semestre do curso os alunos devem cursar a unidade curricular "Circuitos Digitais"; no quarto semestre, os alunos deverão cursar o "Laboratório de Sistemas Computacionais: Circuitos Digitais" e a unidade curricular "Arquitetura e Organização de Computadores"; no quinto semestre, os alunos deverão cursar o "Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores" e a unidade curricular "Linguagens Formais e Autômatos"; no sexto semestre, os alunos cursarão o "Laboratório de Sistemas Computacionais: Engenharia de Sistemas" e a unidade curricular "Compiladores"; no sétimo semestre, os alunos deverão cursar o "Laboratório de Sistemas Computacionais: Compiladores" e a unidade curricular "Sistemas Operacionais"; no oitavo semestre, os alunos cursarão o "Laboratório de Sistemas Computacionais: Sistemas Operacionais" e a unidade curricular "Redes de Computadores" e; por fim, no nono semestre, os alunos realizarão o "Laboratório de Sistemas Computacionais: Redes de Computadores".

Para o cumprimento dessas unidades curriculares integradas, diversas atividades acadêmicas devem ser realizadas para que o sistema computacional possa ser desenvolvido durante o percurso acadêmico dos alunos. De forma mais geral, na Tabela II, encontra-se o fluxo de atividades a ser seguido pelos alunos durante a realização das unidades curriculares integradas mencionadas nos parágrafos anteriores.

Observando essa tabela, pode-se perceber que os alunos são expostos, de forma concomitante e a todo momento, tanto a realização de atividades mais teóricas relacionadas ao aprendizado de conceitos de sistemas computacionais quanto a atividades mais práticas, com o desenvolvimento de determinadas partes do sistema. Além disso, também pode-se perceber que, conforme os alunos vão assimilando um maior conhecimento ao estudar aspectos teóricos diversificados, novas funcionalidades são idealizadas, implementadas e incorporadas ao sistema computacional que está sendo construído, integrando e incorporando-as aos trabalhos anteriormente realizados (reutilização de artefatos).

Na próxima subseção, especificam-se os objetivos de cada unidade curricular mencionada anteriormente, descrevendo-se, com mais detalhes, as diversas atividades teóricas e práticas que devem ser realizadas para o desenvolvimento do sistema computacional.

TABELA II
FLUXO DE ATIVIDADES PARA A REALIZAÇÃO DAS UNIDADES CURRICULARES INTEGRADAS

Termo/ Semestre	Descrição das Principais Atividades	Unidade Curricular
Terceiro	Aprendizado de Circuitos Combinacionais e Sequenciais Projeto de Máquinas de Estados Finitos Estudo da Linguagem de Descrição de Hardware Verilog e de FPGAs Projeto de Circuitos Combinacionais e Sequenciais em Verilog	CD Lab. de CD
Quarto	Estudo sobre Conjunto de Instruções e Modos de Endereçamento Estudo de Memória Cache, Virtual e Mecanismos de Entrada/Saída Estudo sobre Arquiteturas de Computadores Existentes na Literatura Definição do Esquemático da Arquitetura, Conjunto de Instruções e Modos de Endereçamento Implementação em FPGA do processador, memória e interface de Entrada/Saída	 AOC Lab. de AOC
Quinto	Estudo sobre Linguagens e Expressões Regulares e Livre de Contexto Estudo sobre Máquinas de Turing Fundamentação sobre gerenciamento de projetos e visão geral sobre padrões de projeto Aplicação dos conceitos de Engenharia de Sistemas no desenvolvimento de produtos Descrição e Documentação do projeto do sistema computacional em desenvolvimento	 LFA Lab. de ES
Sexto	Estudo sobre Análise Léxica, Sintática e Semântica Fundamentação sobre Geração de Código e Otimização Especificação da Linguagem de Programação para a qual o compilador será construído Modelagem e Implementação do Compilador Aprendizado sobre Gerenciamento de Processo, de Memória, de Entrada/Saída e de Arquivo	 CP Lab. de CP
Sétimo	Estudo sobre Organização, Chamadas de Sistemas e Modularização Estudo de Sistemas Operacionais Reais e Didáticos Definição e Estruturação dos gerenciamentos de processo, memória, Entrada/Saída e Arquivo Implementação do Sistema Operacional definido	SO Lab. de SO
Oitavo	Estudo sobre as camadas de protocolos (física, enlace, rede, transporte, sessão, apresentação e aplicação) Implementação, por meio de chamadas de sistema, de transmissão de sinais	RC
Nono	Implementação de transmissão de dados entre processos (portas de comunicação)	Lab. de RC

Legenda: CD - Circuitos Digitais; Lab. de CD - Laboratório de Sistemas Computacionais: Circuitos Digitais; AOC - Arquitetura e Organização de Computadores; Lab. de AOC - Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores; LFA - Linguagens Formais e Autômatos; Lab. de ES - Laboratório de Sistemas Computacionais: Engenharia de Sistemas; CP - Compiladores; Lab. de CP - Laboratório de Sistemas Computacionais: Compiladores; SO - Sistemas Operacionais; Lab. de SO - Laboratório de Sistemas Computacionais: Sistemas Operacionais; RC - Redes de Computadores e; Lab. de RC - Laboratório de Sistemas Computacionais: Redes de Computadores.

B. Objetivos Gerais e Específicos

1) Circuitos Digitais: A unidade curricular de "Circuitos Digitais" ocorre em quatro horas semanais durante dezoito semanas totalizando uma carga horária de 72 horas e abordando conteúdos mais teóricos que envolvem diversos tópicos, sendo eles: sistemas de numeração; funções lógicas, álgebra booleana e portas lógicas; simplificação de funções booleanas; circuitos combinacionais: conversores, decodificadores, multiplexadores, demultiplexadores e geradores de paridade; circuitos combinacionais aritméticos: somadores, subtratores, multiplicadores e comparadores de magnitude; circuitos sequenciais: latches, flip flops e registradores; máquinas de estados finitos: Moore e Mealy e projeto de circuitos combinacionais e sequenciais.

2) Laboratório de Sistemas Computacionais: Circuitos Digitais: A unidade curricular "Laboratório de Sistemas Computacionais: Circuitos Digitais" ocorre em duas horas semanais durante dezoito semanas, totalizando uma carga horária de 36 horas. Essa unidade curricular aborda conteúdos mais práticos tendo como principais objetivos:

- a descrição de sistemas digitais utilizando níveis de abstração diferentes (porta lógica, transferência entre registradores e comportamental);
- a implementação de circuitos digitais combinacionais utilizando uma linguagem de descrição de hardware;
- a implementação de circuitos digitais sequenciais utilizando uma linguagem de descrição de hardware;

- a realização de simulações e verificação da funcionalidade dos circuitos projetados; e
- a realização de testes e comparação das funcionalidades dos circuitos implementados com os resultados obtidos na simulação.

Para a realização desses objetivos cada aluno possui em sua bancada um kit educacional DE2-115 para a implementação em lógica programável dos circuitos digitais que são propostos durante o semestre.

Os alunos iniciam a unidade curricular com implementações em FPGAs de circuitos combinacionais e sequenciais utilizando desenhos ou esquemáticos produzidos com o software Quartus Prime. Após a familiarização dos alunos com os kits DE2-115 e com o software Quartus Prime, são introduzidos conceitos básicos sobre a linguagem de descrição de hardware Verilog e seus níveis de modelagem: modelagem no nível de portas lógicas, modelagem no nível de transferência entre registradores (RTL) e modelagem no nível comportamental. Por fim, os alunos finalizam a unidade curricular implementando circuitos combinacionais e sequenciais descritos em Verilog por meio da realização de projetos práticos envolvendo circuitos aritméticos e máquinas de estados finitos.

3) Arquitetura e Organização de Computadores: A unidade curricular de ”Arquitetura e Organização de Computadores” ocorre em quatro horas semanais durante dezoito semanas totalizando uma carga horária de 72 horas e abordando conteúdos mais teóricos que envolvem diversos tópicos, sendo eles: organização de computadores: processador, memória, entrada/saída; sistema de memória; componentes da unidade central de processamento: a unidade lógica e aritmética e a unidade de controle; conjunto de instruções; modos de endereçamento; arquitetura RISC (*Reduced Instruction Set Computer*) e CISC (*Complex Instruction Set Computer*); noções de linguagem de máquina; memória cache; pipeline; arquiteturas superescalares; sistema multiprocessado; memória virtual; e mecanismos de entrada/saída.

Esta unidade curricular aborda os conceitos teóricos necessários para o projeto e desenvolvimento de um sistema computacional composto por processadores, memórias e sistemas de entrada/saída. Com isso, após a conclusão dessa unidade curricular, os alunos podem prosseguir para a realização do ”Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores”.

4) Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores: Nessa unidade curricular, que ocorre em 4 horas semanais por dezoito semanas totalizando 72 horas, cada aluno deve desenvolver em Verilog sua própria plataforma de hardware, composta por processador, memória e sistema de entrada/saída utilizando o kit DE2-115 e o software Quartus Prime. Sendo assim, os objetivos principais dessa unidade curricular são:

- a descrição da arquitetura de um processador utilizando uma ferramenta de descrição de hardware;
- a utilização de lógica programável para implementar um processador;
- a realização de simulações e testes para verificar a funcionalidade do sistema projetado;
- o desenvolvimento em lógica programável de um sistema de memória;
- o desenvolvimento em lógica programável de um sistema de comunicação; e
- a elaboração de apresentações orais e redação de textos.

5) Laboratório de Sistemas Computacionais: Engenharia de Sistemas: Nessa unidade curricular, que ocorre em 2 horas semanais por dezoito semanas totalizando 36 horas, cada aluno deve elaborar a especificação do projeto de um sistema computacional complexo, tanto do ponto de vista do software como do hardware. Sendo assim, o objetivo geral dessa unidade curricular é capacitar o aluno a conceber e especificar, em termos sistêmicos, seus projetos de engenharia, tanto no nível de produtos como serviços e negócios, tendo como principais objetivos específicos:

- oferecer ao aluno a fundamentação sobre sistemas e a ciência de sistemas;
- capacitar o aluno a realizar projetos de engenharia baseando-se em conceitos de gerenciamento de projetos;
- capacitar o aluno a conceber, especificar e desenvolver artefatos de engenharia a partir de uma visão integrada de sistemas;
- oferecer ao aluno uma visão geral dos principais padrões de Engenharia de Sistemas;
- capacitar o aluno a aplicar os conceitos de Engenharia de Sistemas no desenvolvimento de produtos, processos e serviços e;
- capacitar o aluno a desenvolver apresentações orais e redação de textos.

6) Linguagens Formais e Autômatos: A unidade curricular de ”Linguagens Formais e Autômatos” ocorre em quatro horas semanais durante dezoito semanas totalizando uma carga horária de 72 horas e abordando conteúdos teóricos que envolvem diversos tópicos, sendo eles: linguagens regulares - autômatos finitos determinísticos e não-determinísticos; expressões regulares; linguagens livres de contexto - gramáticas livres de contexto; autômatos de pilha; linguagens sensíveis ao contexto e linguagens recursivamente enumeráveis - máquinas de turing; tese de Church-Turing; e indecibilidade - máquinas de turing universais.

Esta unidade curricular está relacionada à área de teoria da computação, preparando os alunos para a realização da unidade curricular de ”Compiladores”.

7) Compiladores: A unidade curricular de ”Compiladores” ocorre em quatro horas semanais durante dezoito semanas totalizando uma carga horária de 72 horas e abordando conteúdos teóricos que envolvem diversos tópicos, sendo eles: sistema de varredura - análise léxica; gerador de analisador léxico; análise sintática descendente; análise sintática ascendente; gerador de analisador sintático; análise semântica; geração de código; e otimização de código.

Esta unidade curricular aborda os conceitos teóricos necessários para o projeto e desenvolvimento de um compilador,o qual deverá ser especificado para gerar códigos executáveis para a plataforma de hardware desenvolvida por cada aluno na unidade curricular anterior de ”Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores”.

8) Laboratório de Sistemas Computacionais: Compiladores: Nessa unidade curricular, que ocorre em 4 horas semanais por dezoito semanas totalizando 72 horas, cada aluno deve implementar um compilador completo para o sistema computacional especificado e implementado nos laboratórios integrados anteriores. Sendo assim, o objetivo geral dessa unidade curricular é capacitar o aluno a construir um compilador completo, envolvendo o processo de análise e síntese do compilador. Como objetivos específicos, têm-se:

- capacitar o aluno a especificar a linguagem de programação de alto nível, para a qual o compilador será construído;
- capacitar o aluno na especificação e modelagem do compilador a ser implementado;
- construir os módulos de análise léxica, sintática e semântica do compilador;
- construir os módulos de geração e otimização de código objeto da máquina alvo; e
- capacitar o aluno a desenvolver apresentações orais e redação de textos relativos aos conteúdos trabalhados na unidade curricular.

Para o compilador projetado, códigos fontes da linguagem especificada devem ser automaticamente traduzidos para códigos executáveis na plataforma de hardware desenvolvida pelo aluno na unidade curricular do semestre anterior de ”Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores”. Para aprovação na unidade curricular, o aluno deve demonstrar o correto funcionamento de seu compilador na tradução de códigos fontes para os seus respectivos códigos de máquina, os quais devem ser testados sintetizando a plataforma de hardware no kit FPGA DE2-115 e executando, sobre essa plataforma, os códigos de máquina gerados.

9) Sistemas Operacionais: A unidade curricular de ”Sistemas Operacionais” ocorre em quatro horas semanais durante dezoito semanas totalizando uma carga horária de 72 horas e abordando conteúdos mais teóricos que envolvem diversos tópicos, sendo eles: conceitos sobre processos, organizações de sistemas operacionais e chamadas de sistema; gerência do processador - estados de processo e escalonamento; Entrada e saída - dispositivos e controladores, interrupções, dependência e independência; gerência de memória - partições fixas e variáveis, paginação, segmentação e memória virtual; e gerência de arquivos.

Esta unidade curricular aborda os conceitos teóricos necessários para o projeto e desenvolvimento de um sistema operacional, o qual ocorrerá na próxima unidade curricular integrada de ”Laboratório de Sistemas Computacionais: Sistemas Operacionais”.

10) Laboratório de Sistemas Computacionais: Sistemas Operacionais: Nessa unidade curricular, que ocorre em 4 horas semanais por dezoito semanas totalizando 72 horas, cada aluno deve projetar e implementar um sistema operacional para um sistema digital em lógica programável composto por processador, memória e interface de comunicação, desenvolvido na unidade curricular de ”Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores” de acordo com as especificações e documentação produzidas no ”Laboratório de Sistemas Computacionais: Engenharia de Sistemas”. Além disso, o aluno deverá gerar o código executável do sistema operacional utilizando o compilador desenvolvido na unidade curricular de ”Laboratório de Sistemas Computacionais: Compiladores”. Como objetivos específicos, têm-se:

- estudar os recursos do sistema operacional oferecidos aos programas de usuário (chamadas de sistema);
- estudar sistemas operacionais reais e didáticos, verificando suas características e identificando componentes de software e políticas adotadas;
- definir e implementar um sistema operacional para uma plataforma de hardware específica utilizando um compilador próprio; e
- capacitar o aluno a desenvolver apresentações orais e redação de textos relativos aos conteúdos trabalhados na unidade curricular.

A plataforma de hardware desenvolvida anteriormente deve ser sintetizada no kit FPGA DE2-115 e o sistema operacional projetado deve ser colocado para execução sobre essa plataforma. A aprovação do aluno está condicionada ao correto funcionamento do sistema operacional realizado no Kit.

11) Redes de Computadores: A unidade curricular de ”Redes de Computadores” ocorre em quatro horas semanais durante dezoito semanas totalizando uma carga horária de 72 horas e abordando conteúdos mais teóricos que envolvem diversos tópicos, sendo eles: conceitos gerais sobre medidas de desempenho, camadas de protocolos e serviços; histórico das redes de computadores e internet; camada física - características do meio de transmissão e técnicas de transmissão; camada de aplicação - fundamentos das aplicações de rede e principais protocolos da camada de aplicação; camada de transporte - introdução e serviços da camada de transporte, protocolos TCP (*Transmission Control Protocol*) e UDP (*User Datagram Protocol*) e princípios do controle de congestionamento; camada de rede - o protocolo IPv4 (*Internet Protocol version 4*), o protocolo IPv6 (*Internet Protocol version 6*) e algoritmos de roteamento; camada de enlace e redes locais - serviços oferecidos pela camada de enlace, protocolos de acesso múltiplo, endereçamento na camada de enlace e redes Ethernet; redes sem fio; redes móveis; e princípios da gerência de redes.

Esta unidade curricular tem como objetivo possibilitar que os alunos adquiram conhecimentos sólidos sobre os principais conceitos e desafios relacionados às arquiteturas, serviços e protocolos das redes de computadores, habilitando-os desenvolver e implementar, na unidade curricular do próximo semestre de ”Laboratório de Sistemas Computacionais: Redes de Computadores”, uma chamada de sistema de comunicação em rede para o sistema operacional desenvolvido anteriormente nos laboratórios integrados.

12) Laboratório de Sistemas Computacionais: Redes de Computadores: Nessa unidade curricular, que ocorre em 2 horas semanais por dezoito semanas totalizando 36 horas, cada aluno deve estender as funcionalidades do sistema operacional projetado no semestre anterior, incorporando o tratamento do gerenciamento de redes, possibilitando a comunicação entre dois ou mais sistemas computacionais. Como objetivos específicos, têm-se:

- implementação de transmissão de sinais digitais por meio de uma interface digital de comunicação;
- implementação de um protocolo de roteamento e encaminhamento de pacotes;
- implementação de um protocolo de transporte; e
- implementação de uma interface de controle e gerenciamento de transmissão de dados entre processos (gerenciamento de portas de comunicação).

Todo o sistema de comunicação especificado e projetado deve ser testado nas plataformas de hardware e software desenvolvidas pelo aluno nos semestres anteriores, devendo mapeá-los no kit FPGA DE2-115. A aprovação do aluno está condicionada ao correto funcionamento de todo o sistema computacional realizado no Kit.

C. Metodologia de Ensino-Aprendizagem Elaborada para os Laboratórios Integrados

Uma metodologia de ensino-aprendizagem específica foi elaborada, sendo empregada nos laboratórios integrados visando a realização do projeto do sistema computacional e possibilitando a integração entre as unidades curriculares.

A metodologia proposta consiste da execução consecutiva de várias etapas de realização de tarefas em direção à implementação das especificações de projeto estabelecidas, onde laços iterativos contendo um conjunto dessas etapas são realizados diversas vezes no processo de ensino-aprendizagem. Na Fig. 2 encontra-se o fluxograma da metodologia de ensino elaborada e empregada nos laboratórios integrados.

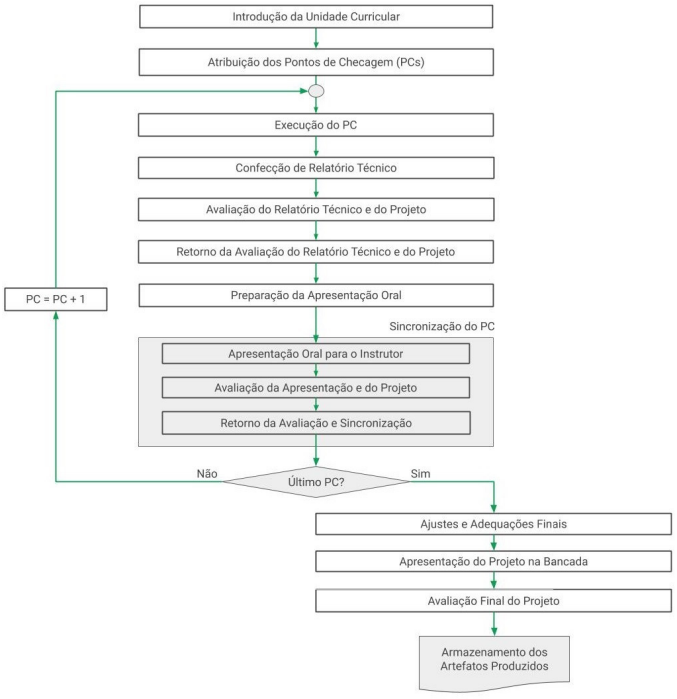

Fig. 2. Fluxograma da metodologia de ensino-aprendizagem elaborada para os laboratórios integrados.

De acordo com a Fig. 2, treze etapas de realização de tarefas são distribuídas ao longo da aplicação da metodologia, tendo sua iteratividade centrada em relação à quantidade de Pontos de Checagem (PCs). Como definição, os pontos de checagem podem ser considerados metas ou objetivos intermediários que precisam ser realizados passo-a-passo pelos alunos para que, no final, a implementação das especificações estabelecidas possa ser alcançada.

Tomando como exemplo o caso da unidade curricular de ”Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores”, são definidos quatro PCs, sendo eles: no primeiro ponto de checagem (PC1) os alunos são estimulados a pesquisar o funcionamento e arquitetura interna de vários processadores da literatura e a começarem a definir o conjunto de instruções (ISA) que será suportado pela sua plataforma de hardware; no segundo ponto de checagem (PC2) o aluno deve realizar um detalhamento maior do esboço da arquitetura interna e projetar em Verilog a unidade de processamento da plataforma de hardware; no terceiro ponto de checagem (PC3) o aluno deve realizar a especificação em Verilog da unidade de controle da plataforma de hardware proposta e, por fim, no quarto e último ponto de checagem (PC4) o aluno deve trabalhar na integração de todos os módulos produzidos, interligando a unidade de processamento e de controle da plataforma de hardware.

Num segundo exemplo, para o caso da unidade curricular de ”Laboratório de Sistemas Computacionais: Sistemas Operacionais”, são definidos três PCs, sendo eles: no primeiro ponto de checagem (PC1), o aluno é estimulado a estudar o gerenciamento de processos, de memória, de sistemas de arquivos e de dispositivos de E/S e, baseado nesse estudo, realizar a definição do sistema operacional a ser projetado, descrevendo as técnicas e algoritmos a serem utilizados; no segundo ponto de checagem (PC2), o aluno deve realizar inclusões e adaptações necessárias em sua plataforma de hardware para o sistema operacional a ser executado, além de realizar simulações e testes na plataforma de hardware para comprovar a sua funcionalidade e; por fim, no terceiro ponto de checagem (PC3), o aluno deve realizar a implementação das técnicas e algoritmos descritos, buscando integrá-los à plataforma de hardware.

Como observado no fluxograma da Fig. 2, a metodologia de ensino é composta pela realização consecutiva de várias etapas de realização de tarefas, sendo assim especificadas:

- **Introdução da Unidade Curricular:** na primeira etapa referente à aplicação da metodologia de ensino proposta, o instrutor deverá explicar o plano de ensino da unidade curricular, definindo claramente seus objetivos, ementa, conteúdo programático, bibliografias recomendadas e o funcionamento de forma detalhada da metodologia de ensino que será empregada.
- **Atribuição dos Pontos de Checagem (PCs):** na segunda etapa o instrutor deverá definir, anunciar e explicar aos alunos os pontos de checagem (PCs) estabelecidos para a unidade curricular, os quais irão direcionar o processo de aprendizagem dos alunos.
- **Execução do PC:** na terceira etapa, as metas e objetivos delineados no PC em questão devem buscar ser completados pelos alunos por meio da realização de atividades devidamente definidas e programadas para o desenvolvimento do projeto.

- **Confecção de Relatório Técnico:** na quarta etapa, os alunos devem redigir um relatório técnico sobre a execução do PC correspondente.
- **Avaliação do Relatório Técnico e do Projeto:** na quinta etapa, o instrutor deverá avaliar os relatórios dos alunos e o projeto que está sendo especificado e desenvolvido.
- **Retorno da Avaliação do Relatório Técnico e do Projeto:** na sexta etapa, o instrutor deverá, baseando-se nos projetos e seus respectivos relatórios, descrever os pontos fortes e fracos e discutir possíveis melhorias ou propor novos caminhos a serem seguidos.
- **Preparação da Apresentação Oral:** na sétima etapa, os alunos irão confeccionar o conteúdo visual que será utilizado em suas apresentações orais, devendo-os versar sobre o cumprimento do PC em questão.
- **Apresentação Oral para o Instrutor:** na oitava etapa, os alunos realizarão a apresentação oral para o instrutor da unidade curricular, utilizando-se, para isso, os recursos instrucionais necessários, como por exemplo, quadro branco, computadores, kits FPGAs DE2-115, projetores multimídia, entre outros.
- **Avaliação da Apresentação e do Projeto:** na nona etapa, o instrutor avaliará as apresentações orais dos alunos e o desenvolvimento do projeto em relação às especificações estabelecidas.
- **Retorno da Avaliação e Sincronização:** na décima etapa, o instrutor deverá realizar um retorno formativo aos alunos sobre o conteúdo apresentado e sobre o andamento do projeto, devendo indicar, caso necessário, as atividades que ainda precisarão ser realizadas para o cumprimento dos objetivos esperados.
- **Ajustes e Adequações Finais:** na décima primeira etapa, os alunos deverão realizar os ajustes e adequações finais em relação ao desenvolvimento do projeto, devendo levar em considerações as avaliações e retornos formativos produzidos pelo instrutor da unidade curricular.
- **Apresentação do Projeto na Bancada:** na décima segunda etapa, os alunos deverão apresentar na bancada o projeto realizado, demonstrando o correto funcionamento do sistema construído.
- **Avaliação Final do Projeto:** por fim, na décima terceira e última etapa da metodologia, o instrutor deverá realizar a avaliação de todo o projeto construído levando em consideração os objetivos da unidade curricular em questão, determinando-se, inclusive, o conceito final dos alunos.

Para a finalização da unidade curricular, é necessário o armazenamento institucional dos artefatos gerados durante a realização dos pontos de checagem. O objetivo desse armazenamento é permitir que, na unidade curricular subsequente, os alunos possam continuar o desenvolvimento do sistema computacional reutilizando, para isso, os artefatos anteriormente produzidos.

Atualmente, tem-se utilizado o ambiente virtual de aprendizagem Moodle[2] como repositório dos artefatos que estão sendo produzidos pelos alunos ao longo de suas trajetórias acadêmicas. Esse ambiente de aprendizagem é utilizado para suportar a aplicação da metodologia de ensino-aprendizagem comentada nos parágrafos anteriores, sendo organizado para permitir a realização, o mapeamento e a submissão das atividades propostas em cada ponto de checagem.

IV. Qualidade Educacional de Ensino Obtida

Durante o progresso do aluno no curso, os laboratórios integrados foram medidos por meio de um questionário de avaliação sobre a qualidade educacional ou efetividade do ensino aplicado. O questionário foi preenchido eletronicamente de forma anônima e não-obrigatória, tendo sido publicado no Moodle por uma semana durante o término das unidades curriculares de laboratórios de sistemas computacionais.

O questionário aplicado baseia-se no instrumento multidimensional de avaliação do ensino mundialmente conhecido como *Students' Evaluation of Educational Quality* (SEEQ) [26]. O SEEQ, além de ser utilizado em diversas universidades no mundo e ter propriedades psicométricas bem conhecidas [27], tem sua validade e reprodutibilidade sido confirmadas internacionalmente em diversos países [28].

No questionário aplicado, apresentado na Tabela III, os alunos indicam seu grau de concordância numa escala de resposta *likert* de 5 pontos sobre 32 afirmações agrupadas em nove dimensões relacionadas à efetividade do ensino, sendo eles: aprendizado, entusiasmo, organização, interação com o grupo, empatia, amplitude na abordagem, processo de avaliação, atividades/atribuições e carga de trabalho/ dificuldade.

Para as oito primeiras dimensões a estratificação da escala segue o padrão: 1 correspondendo a "discordo plenamente", 2 correspondendo a "discordo", 3 correspondendo a "não concordo nem discordo", 4 correspondendo a "concordo" e 5 correspondendo a "concordo plenamente". Para a última dimensão, a estratificação da escala é definida em conjunto com a afirmação realizada, sendo que a estratificação de maior valor numérico indica uma maior carga de trabalho ou dificuldade encontrada durante a realização da unidade curricular. Além disso, o questionário também possui uma questão em aberto, permitindo que o aluno descreva os pontos positivos que considerou ter sido relevantes para o seu aprendizado e os pontos que precisam ser melhorados.

Visando-se reduzir ameaças à validação da proposta de trabalho, houve uma preocupação quando da aplicação do questionário para que uma quantidade razoável de alunos realizassem o seu preenchimento e, ao mesmo tempo, para que os alunos não se sentissem pressionados em atribuir determinadas estratificações por desconfiarem da possibilidade de ocorrência de penalizações decorrentes de suas respostas. Para reduzir essas ameaças, três medidas foram realizadas: (1) antes da disponibilização do questionário, a coordenação

[2]O Moodle (*Modular Object-Oriented Dynamic Learning Environment*) é uma plataforma computacional de código aberto utilizado como apoio à aprendizagem do aluno. Mais detalhes sobre essa plataforma podem ser encontrados no endereço eletrônico https://moodle.org, acesso em 14 jun. 2019.

TABELA III
QUESTIONÁRIO DE AVALIAÇÃO SOBRE A QUALIDADE EDUCACIONAL APLICADO AOS ALUNOS

Adaptado de [26]

Dimensão	Num.	Questão/Afirmação
Aprendizado (***Learning***)	1	Você considerou a unidade curricular (UC) intelectualmente desafiadora e estimulante
	2	Você aprendeu algo que considera importante
	3	Seu interesse no conteúdo aumentou como consequência desta UC
	4	Você aprendeu e entendeu os materiais de apoio (*slides*, livros, etc.) desta UC
Entusiasmo (***Enthusiasm***)	5	O professor era entusiasmado em relação ao ensinamento dos conteúdos curriculares da UC
	6	O professor foi dinâmico e enérgico na condução da UC
	7	A UC foi realizada com simpatia e bom senso pelo professor
	8	O modo como o professor ministrou a UC manteve seu interesse durante todo o período
Organização (***Organisation***)	9	As explicações do professor eram claras
	10	Os materiais didáticos da UC estavam bem preparados e foram cuidadosamente explicados
	11	Os objetivos apresentados estavam em consonância com aqueles realmente ensinados, permitindo-me entender o percurso da UC
	12	O planejamento da UC realizado pelo professor facilitou a assimilação do conteúdo
Interação com o Grupo (***Group Interaction***)	13	Os alunos foram incentivados a contribuir com outros alunos da turma durante a realização da UC
	14	Os alunos foram estimulados a compartilhar suas ideias, conhecimentos ou pontos de vista
	15	Os alunos foram encorajados a fazer perguntas e receberam respostas relevantes
	16	Os alunos foram incentivados a expressar suas próprias ideias
Empatia (***Individual Rapport***)	17	O professor era amistoso e cordial com os alunos, individualmente
	18	O professor fazia os alunos sentirem-se bem vindos ao pedir ajuda e conselho, dentro ou fora da aula
	19	O professor tinha um interesse sincero nos alunos, individualmente
	20	O professor era adequadamente acessível aos alunos durante a realização da UC
Amplitude na Abordagem (***Breadth***)	21	Em relação ao conteúdo, foi possível identificar e comparar as implicações de várias teorias
	22	O contexto ou a origem dos conceitos foram trabalhados na UC
	23	Foram apresentados outros pontos de vista além do seu, quando apropriado
	24	Com a realização da UC, os avanços na área puderam ser estabelecidos
Processo de Avaliação (***Examinations***)	25	Os comentários do professor e as avaliações recebidas foram importantes
	26	Os métodos de avaliação eram justos e apropriados
	27	Os conteúdos das atividades de avaliação estavam de acordo com os enfatizados pelo professor
Atividades/Atribuições (***Assignments***)	28	O material de estudo e/ou bibliografia recomendados foram importantes
	29	As leituras, trabalhos de pesquisa, tarefas realizadas, etc. contribuíram para a compreensão e apreciação dos conteúdos curriculares da UC
Carga de Trabalho/ Dificuldade (***Overall***)	30	A dificuldade desta UC, em relação às outras UCs, foi: (1- Muito fácil; 2 - fácil; 3 - Média; 4 - Difícil; 5 - Muito difícil)
	31	A exigência/carga de trabalho desta UC, em relação às outras UCs foi: (1 - Muito leve; 2 - Leve; 3 - Média; 4 - Pesada; 5 - Muito pesada)
	32	O ritmo de andamento da UC foi: (1 - Muito lento; 2 - Lento; 3 - Médio; 4 - Rápido; 5 - Muito rápido)
Comentários (Questão em Aberto)	33	Por favor, descreva os pontos positivos que você considera ter sido relevantes para o aprendizado da disciplina e os pontos que precisam ser melhorados

de curso conversou com os alunos sobre os motivos para a sua aplicação, enfatizando que os resultados produzidos seriam utilizados, única e exclusivamente, com o intuito de se identificar problemas na realização dos laboratórios integrados buscando solucioná-los para a implantação ou estabelecimento de um curso de Engenharia de Computação melhor; (2) foi informado, por meio do ambiente virtual de aprendizagem Moodle, que o preenchimento seria de caráter anônimo, sem nenhuma identificação individual dos alunos e; (3) a aplicação ocorreu somente após o término da unidade curricular e o fechamento de notas para que os alunos se sentissem livres e inferissem a não existência da possibilidade de algum tipo de retaliação na unidade curricular.

Para se ter uma visão geral do comportamento dos laboratórios integrados em relação à qualidade educacional medida, na Fig. 3 e na Fig. 4 encontram-se os resultados obtidos com a aplicação do questionário SEEQ no primeiro e último anos da realização dos projetos de sistemas computacionais que estão sendo desenvolvidos pelos alunos.

Na Fig. 3 apresentam-se os resultados obtidos com a aplicação do questionário para os alunos da unidade curricular de Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores e, na Fig.4, apresentam-se os resultados obtidos para a unidade curricular de Laboratório de Sistemas Computacionais: Sistemas Operacionais.

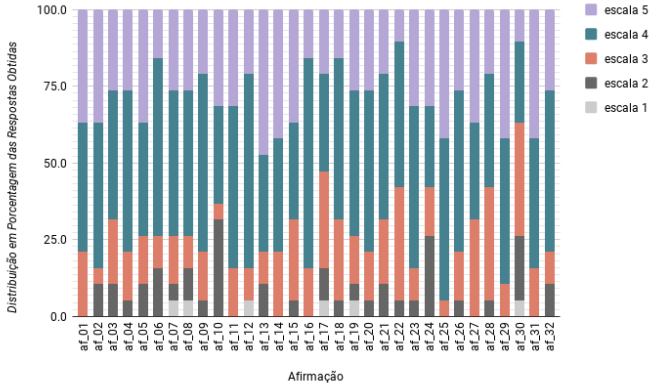

Fig. 3. Respostas obtidas com o preenchimento do questionário SEEQ pelos alunos para o Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores.

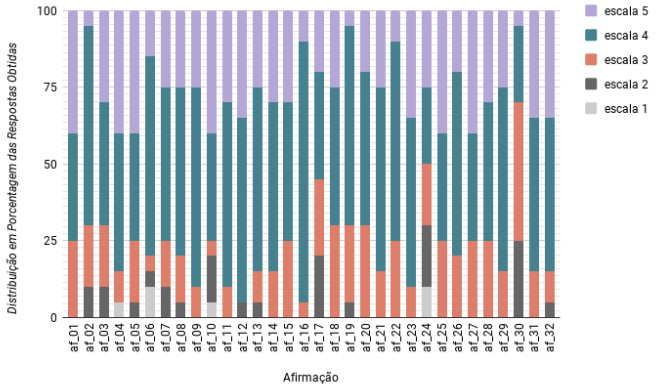

Fig. 4. Respostas obtidas com o preenchimento do questionário SEEQ pelos alunos para o Laboratório de Sistemas Computacionais: Sistemas Operacionais.

Para a unidade curricular de Laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores foram computadas 19 respostas de alunos, enquanto que para a unidade curricular de Laboratório de Sistemas Computacionais: Sistemas Operacionais foram computadas 20 respostas. O número de vagas de entrada no curso de Engenharia de Computação é de 25 alunos. A diferença entre a quantidade de respostas obtidas e o número de vagas de entrada refere-se a diversos fatores, sendo os principais: desistência na unidade curricular (reprovação do aluno por frequência), abandono do curso, não preenchimento do questionário disponibilizado e reprovações em pré-requisitos. No entanto, vale observar que a quantidade de respostas obtidas foi significativa, sendo próxima a 80% de representatividade dos alunos. Além disso, o aluno reprovado num determinado laboratório integrado precisará, num próximo momento, se matricular novamente na unidade curricular. Por sua vez, o aluno que não obteve êxito na realização dos pré-requisitos correspondentes precisará, primeiramente, ser aprovado nessas unidades curriculares para

poder se matricular no laboratório integrado. Embora os alunos possam trabalhar em seus sistemas computacionais em qualquer momento do curso, a aplicação da metodologia de ensino-aprendizagem descrita na subseção III-C e o respectivo processo de avaliação dos sistemas computacionais que estão sendo desenvolvidos ocorrerão apenas quando da inscrição dos alunos na unidade curricular do laboratório integrado correspondente.

Como exemplo de interpretação dos gráficos apresentados tem-se que, na Fig. 3, a composição de respostas dos alunos para a afirmação 07 do questionário SEEQ foi de 5,3% na escala 1, 5,3% na escala 2, 15,8% na escala 3, 47,4% na escala 4 e 26,3% na escala 5.

Observando cronologicamente a Fig. 3 e a Fig. 4 em relação às afirmações 01 e 03, pode-se perceber que os laboratórios integrados tem tido a capacidade de manter os alunos motivados e entusiasmados ao longo de suas trajetórias acadêmicas em relação ao aprendizado de sistemas computacionais.

No primeiro ano de desenvolvimento do sistema computacional (Fig. 3), a afirmação 01 obteve uma média geral de 4,16 pontos na escala *likert*, correspondendo a uma composição de aproximadamente 21% das respostas na escala 3, 42% das respostas na escala 4 e 37% na escala 5; para a afirmação 03, obteve-se uma média geral de 3,84 pontos na escala *likert*, correspondendo a uma composição de aproximadamente 69% nas escalas 4 e 5.

Essa motivação e entusiasmo dos alunos relatados se mantiveram durante seus percursos acadêmicos, como pode ser constatado no último ano de desenvolvimento do sistema computacional (Fig. 4), em que se obteve uma média geral de 4,15 pontos na escala *likert* para a afirmação 01, correspondendo a uma composição de aproximadamente 74% das respostas nas escalas 4 e 5; para a afirmação 03, obteve-se uma média geral de 3,90 pontos na escala *likert*, correspondendo a uma composição de aproximadamente 70% nas escalas 4 e 5.

Uma consequência interessante referente aos laboratórios integrados é a necessidade de adoção de metodologias de ensino não tradicionais nas unidades curriculares correspondentes para que os alunos possam acompanhar, avaliar e regular suas próprias aprendizagens, buscando-se assim uma abordagem centrada no aluno, que o leve à pesquisa e à produção de conhecimento.

Dentro deste contexto, para promover a colaboração e cooperação entre os alunos no desenvolvimento de seus sistemas computacionais, os instrutores responsáveis por essas unidades curriculares têm empregado práticas de ensino diferenciadas, fazendo-se uso, por exemplo, de avaliações por pares [29] [30] e de rubricas [31] [32] como métodos de ensino-aprendizagem.

O uso dessas práticas de ensino tem levado os alunos a aprimorarem seus projetos, como reportado no trabalho publicado em Oliveira *et al.* [33], e a melhorarem suas habilidades de escrita, como descrito em Oliveira et al. [34].

Como reflexo dessa combinação necessária entre os laboratórios integrados e o uso de metodologias ativas de ensino-aprendizagem, pode-se perceber os resultados positivos reportados pelos alunos quanto à liberdade no desenvolvimento de seus projetos e o compartilhamento de ideias, conhecimentos e pontos de vista (afirmações 14 e 16). Levando em consideração tanto as respostas reportadas na Fig. 3 como na Fig. 4, a afirmação 14 obteve uma média de 4,18 pontos na escala *likert*, enquanto a afirmação 16 obteve 4,02 pontos na escala *likert*.

Em cada laboratório integrado, os instrutores buscam avaliar o progresso do aluno no desenvolvimento do sistema computacional. Para isso, durante vários momentos de realização de uma determinada unidade curricular integrada são avaliados apresentações orais, relatórios técnicos, simulações e testes parciais, apresentações nas bancadas e os artefatos que estão sendo produzidos. Essa diversidade de avaliações e retornos formativos realizados ao longo de todo o semestre letivo produziu resultados muito satisfatórios no questionário SEEQ aplicado aos alunos em relação à dimensão de processo de avaliação cuja média geral foi de aproximadamente 4,12 pontos na escala *likert*.

No entanto, embora os laboratórios integrados tenham trazido resultados positivos significativos, como comentado nos parágrafos anteriores, também é possível perceber que a exigência e a carga de trabalho percebidos pelos alunos para a realização de todo o sistema computacional é alto, se comparado com outras unidades curriculares do curso (afirmação 31). No laboratório de Sistemas Computacionais: Arquitetura e Organização de Computadores, a média obtida foi de 4,26 pontos na escala *likert*, mantendo-se alta até o último ano de desenvolvimento do sistema computacional, representado pelo Laboratório de Sistemas Computacionais: Sistemas Operacionais, com uma média de aproximadamente 4,20 pontos na escala *likert*.

Após o término do último Laboratório de Sistemas Computacionais: Redes de Computadores, os alunos foram convidados para uma conversa sobre a estrutura curricular integrada adotada no curso e as dificuldades encontradas durante a realização do sistema computacional.

De modo geral, os alunos consideraram o Laboratório de Sistemas Computacionais: Compiladores como sendo o mais difícil, seguido do Laboratório de Sistemas Computacionais: Sistemas Operacionais. Como principal motivo, o tempo disponibilizado no sexto termo do curso para o projeto do compilador foi apontado pelos alunos como insuficiente, exigindo uma dedicação extraclasse maior do que em outras unidades curriculares.

Para o Laboratório de Sistemas Computacionais: Sistemas Operacionais, a maior dificuldade apontada pelos alunos foi em relação ao co-projeto de hardware e software e a falta de material didático específico que pudesse auxiliá-los nessa correlação. Vale observar que as especificações e escolhas de projeto realizadas pelo aluno para o sistema operacional a ser implementado implica adaptações e inclusões de novas funcionalidades na plataforma de hardware desenvolvida anteriormente. Por exemplo, se o aluno optar por implementar o algoritmo round-robin no gerenciamento de processos do seu sistema operacional, a sua plataforma de hardware deverá ser adaptada para conter registradores temporizadores específicos capazes de realizar o cômputo da fração de tempo (*time slice*) de um determinado processo.

V. Considerações Finais

Para evitar a visão fragmentada de um sistema computacional complexo decorrente dos tradicionais currículos de curso de Engenharia de Computação, apresentou-se, neste artigo, uma proposta de abordagem curricular visando a integração entre teoria e prática e entre hardware e software. A abordagem curricular se estrutura em unidades curriculares integradas, onde a cada semestre o aluno deverá desenvolver uma parte de um sistema computacional complexo, finalizando-o após três anos de curso durante a sua trajetória acadêmica.

Diferentemente das abordagens apresentadas nos trabalhos relacionados da seção II, a estrutura curricular proposta neste artigo proporciona, como característica principal, a transversalidade de projeto, além de possibilitar que diversos conteúdos relacionados a sistemas computacionais sejam constantemente *revisitados* pelos alunos.

A título de exemplo, um aluno que esteja matriculado na unidade curricular de "Laboratório de Sistemas Computacionais: Sistemas Operacionais" no oitavo termo terá que realizar adaptações em sua plataforma de hardware para implementar registradores temporizadores que possibilitem o gerenciamento de processos. Dessa forma, embora os conteúdos relacionados a Circuitos Digitais, Verilog e Arquitetura e Organização de Computadores estejam previstos para serem aprendidos no terceiro e quarto termos do currículo do curso, o aluno terá que revisitá-los novamente no oitavo termo para conseguir projetar o sistema operacional que está sendo construído, implicando, portanto, num aprendizado mais contínuo.

Além do exposto nos parágrafos anteriores, também pode-se citar, como contribuição do trabalho relatado neste artigo, a metodologia de ensino-aprendizagem elaborada para os laboratórios integrados. A metodologia especificada visa a realização do projeto do sistema computacional complexo e possibilita a integração entre as unidades curriculares.

Com a adoção da estrutura curricular e da metodologia de ensino-aprendizagem propostas neste artigo, os alunos tem se mostrado constantemente motivados e entusiasmados em relação ao aprendizado de sistemas computacionais (afirmações 01 e 03 dos questionários SEEQs preenchidos pelos alunos), além de estar imprimindo habilidades e atitudes que vão além da reprodução de experimentos, incentivando-os à colaboração (afirmação 13) e ao pensamento criativo e inovador (afirmações 14 e 16).

Em relação à motivação e entusiasmo empregados pelos alunos, vale comentar a atitude curiosa que ocorreu no término da última unidade curricular integrada, no Laboratório de Sistemas Computacionais: Redes de Computadores. Alguns alunos, após as apresentações finais nas bancadas, pediram para tirar fotos junto aos instrutores e ao produto final desenvolvido para guardarem de lembrança.

As unidades curriculares propostas na seção III e seus respectivos laboratórios legitimam os conceitos, fundamentos e conteúdos apresentados durante a trajetória acadêmica do aluno ao serem estruturados em torno de um mesmo propósito: a construção ou projeto de um sistema computacional complexo. Com isso, ocorre, portanto, a articulação e interação de diversos conhecimentos de forma recíproca e também coordenada, integrando resultados em busca de uma solução para o propósito estabelecido. Neste sentido, transcende-se à abordagem tradicional curricular multidisciplinar, indo em direção a um contexto educacional mais inovador e moderno voltado à interdisciplinaridade.

Devido à interdisciplinaridade intrínseca comentada no parágrafo anterior e a sua correspondente e constante necessidade de comunicação entre os instrutores responsáveis pelos oferecimentos das unidades curriculares integradas, o projeto pedagógico do curso prevê a criação da função denominada coordenador dos laboratórios de sistemas computacionais. Para cada turma ingressante, a comissão de curso da Engenharia de Computação indica um docente que atuará como coordenador dos laboratórios de sistemas computacionais e acompanhará a turma em todas as unidades curriculares que compõem o eixo de desenvolvimento desse sistema. Este coordenador trabalhará junto aos instrutores das unidades curriculares para a definição, planejamento e acompanhamento das atividades a serem realizadas em cada semestre, visando à integração e o correto funcionamento do sistema durante a realização de todo o projeto.

Para melhoria dos laboratórios integrados, tem-se atuado, como trabalhos futuros, em três frentes. Numa primeira frente, visando reduzir os apontamentos realizados pelos alunos, está se trabalhando na elaboração de materiais de apoio específicos para os laboratórios integrados, buscando enfatizar o co-projeto de hardware e software e a integração entre teoria e a prática, oferecendo documentos e tutoriais variados e aderentes com a proposta de desenvolvimento de um sistema computacional. Numa segunda frente, tem-se buscado incentivar os alunos no aprimoramento de seus projetos com a inclusão de oficinas específicas cujos artefatos produzidos possam ser incorporados posteriormente no desenvolvimento de seus sistemas computacionais. Por exemplo, numa dessas oficinas, referentes ao aprendizado de comunicação digital, os alunos aprendem a projetar um circuito eletrônico capaz de enviar e receber informações por meio de um canal de comunicação sem fio. Após a realização dessa oficina, o aluno poderá anexar ou incorporar esse circuito eletrônico em seu projeto do sistema computacional, utilizando-o, por exemplo, na unidade curricular de "Laboratório de Sistemas Computacionais: Redes de Computadores" para a comunicação sem fio entre dois ou mais sistemas computacionais. Por fim, numa terceira frente, devido à demanda elevada, um laboratório remoto de FPGA está sendo projetado na universidade para permitir que os alunos possam trabalhar em seus sistemas computacionais sem a necessidade de estarem fisicamente presentes num laboratório, disponibilizando acesso remoto a um laboratório real de FPGA em 24 horas do dia durante os 7 dias da semana. Sendo assim, os alunos poderão utilizar o laboratório fisicamente durante as aulas e remotamente em horários extraclasse.

Referências

[1] A. F. Zorzo, D. Nunes, E. S. Matos, I. Steinmacher, J. C. Leite, R. Araujo, R. C. M. Correia, and S. Martins, *Referenciais de Formação para os Cursos de Graduação em Computação*. Sociedade Brasileira de Computação (SBC), 2017.

[2] ACM and IEEE, "Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering," Association for Computing Machinery and IEEE Computer Society, Tech. Rep., 2016.
[3] Unifesp, "Projeto Pedagógico do Curso de Graduação do Bacharelado em Engenharia de Computação," 2015. [Online]. Available: http://www.unifesp.br/campus/sjc/o-curso-engcom/projeto-pedagogico-do-curso.html
[4] C. M. Kellett, "A project-based learning approach to programmable logic design and computer architecture," *IEEE Transactions on Education*, 2012.
[5] M. D. L. Á. Cifredo-Chacõn, Á. Quirõs-Olozábal, and J. M. Guerrero-Rodríguez, "Computer architecture and FPGAs: A learning-by-doing methodology for digital-native students," *Computer Applications in Engineering Education*, 2015.
[6] E. Larraza-Mendiluze, N. Garay-Vitoria, J. I. Martin, J. Muguerza, T. Ruiz-Vazquez, I. Soraluze, J. F. Lukas, and K. Santiago, "Game-Console-Based Projects for Learning the Computer Input/Output Subsystem," *IEEE Transactions on Education*, vol. 56, no. 4, pp. 453–458, nov 2013. [Online]. Available: http://ieeexplore.ieee.org/document/6502274/
[7] E. Larraza-Mendiluze, N. Garay-Vitoria, I. Soraluze, J. Martín, J. Muguerza, and T. Ruiz-Vázquez, "Using a Real Bare Machine in a Project-Based Learning Environment for Teaching Computer Structure: An Analysis of the Implementation Following the Action Research Model," *Transactions on Computing Education*, 2016.
[8] E. A. Billard, "Introducing software engineering developments to a classical operating systems course," *IEEE Transactions on Education*, 2005.
[9] F. Ortin, D. Zapico, and J. M. Cueva, "Design patterns for teaching type checking in a compiler construction course," *IEEE Transactions on Education*, 2007.
[10] D. Baldwin, "A compiler for teaching about compilers," *ACM SIGCSE Bulletin*, 2003.
[11] M. Mernik and V. Žumer, "An educational tool for teaching compiler construction," *IEEE Transactions on Education*, 2003.
[12] D. Kundra and A. Sureka, "Application of Case-Based Teaching and Learning in Compiler Design Course," *ArXiv e-prints*, 2016. [Online]. Available: http://adsabs.harvard.edu/abs/2016arXiv161100271K
[13] ——, "An Experience Report on Teaching Compiler Design Concepts Using Case-Based and Project-Based Learning Approaches," in *Proceedings - IEEE 8th International Conference on Technology for Education, T4E 2016*, 2017.
[14] A. Garmpis, "Alg-OS-A web-based software tool to teach page replacement algorithms of operating systems to undergraduate students," *Computer Applications in Engineering Education*, 2013.
[15] B. S. Jong, C. H. Lai, Y. T. Hsia, T. W. Lin, and C. Y. Lu, "Using game-based cooperative learning to improve learning motivation: A study of online game use in an operating systems course," *IEEE Transactions on Education*, 2013.
[16] W. K. Wong, "Lessons from adopting a maker approach to teaching operating systems with Raspberry Pi," *Interactive Technology and Smart Education*, 2018.
[17] M. W. El-Kharashi, G. Darling, B. Marykuca, and G. C. Shoja, "Understanding and implementing computer network protocols through a lab project," *IEEE Transactions on Education*, 2002.
[18] N. I. Sarkar, "Teaching computer networking fundamentals using practical laboratory exercises," *IEEE Transactions on Education*, 2006.
[19] S. Winarno, "Students' Feedback of mDPBL Approach and the Learning Impact towards Computer Networks Teaching and Learning," *International Journal of Educational Methodology*, 2018.
[20] K. Abe, T. Tateoka, M. Suzuki, Y. Maeda, K. Kono, and T. Watanabe, "An integrated laboratory for processor organization, compiler design, and computer networking," *IEEE Transactions on Education*, 2004.
[21] D. M. B. Santos and C. A. S. Silva, "Problem-based learning in a computer engineering program: Quantitative evaluation of the students' perspective," *IEEE Latin America Transactions*, 2018.
[22] M. Somerville, D. Anderson, H. Berbeco, J. R. Bourne, J. Crisman, D. Dabby, H. Donis-Keller, S. S. Holt, S. Kerns, D. V. Kerns, R. Martello, R. K. Miller, M. Moody, G. Pratt, J. C. Pratt, C. Shea, S. Schiffman, S. Spence, L. A. Stein, J. D. Stolk, B. D. Storey, B. Tilley, B. Vandiver, and Y. Zastavker, "The olin curriculum: Thinking toward the future," *IEEE Transactions on Education*, 2005.
[23] H. U. Rehman, R. A. Said, and Y. Al-Assaf, "An integrated approach for strategic development of engineering curricula: Focus on students' design skills," *IEEE Transactions on Education*, 2009.
[24] M. Rashid and I. A. Tasadduq, "Holistic development of computer engineering curricula using Y-chart methodology," *IEEE Transactions on Education*, 2014.
[25] Olin College, "Electrical and Computer Engineering (ECE)," 2018. [Online]. Available: http://olin.smartcatalogiq.com/en/2018-19/Catalog/Programs-of-Study-and-Degree-Requirements/Academic-Programs/Electrical-and-Computer-Engineering-ECE
[26] H. W. Marsh, "SEEQ: A reliable, valid, and useful instrument for collection students' evaluations of university teaching," *Br. J. educ. Psychol*, 1982.
[27] M. Coffey and G. Gibbs, "The evaluation of the student evaluation of educational quality questionnaire (SEEQ) in UK higher education," *Assessment and Evaluation in Higher Education*, 2001.
[28] J. T. Richardson, "Instruments for obtaining student feedback: A review of the literature," 2005.
[29] T. Tenório, I. I. Bittencourt, S. Isotani, and A. P. Silva, "Does peer assessment in on-line learning environments work? A systematic review of the literature," pp. 94–107, 2016.
[30] K. Topping, "Self and Peer Assessment in School and University: Reliability, Validity and Utility," in *Optimising New Modes of Assessment: In Search of Qualities and Standards*. Dordrecht: Kluwer Academic Publishers, 2006, pp. 55–87. [Online]. Available: http://link.springer.com/10.1007/0-306-48125-1_4
[31] Y. M. Reddy and H. Andrade, "A review of rubric use in higher education," *Assessment & Evaluation in Higher Education*, vol. 35, no. 4, pp. 435–448, jul 2010. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/02602930902862859
[32] M. E. Huba and J. E. Freed, *Learner-centered Assessment on College Campuses: Shifting the Focus from Teaching to Learning*, 8th ed. Allyn and Bacon, 2000. [Online]. Available: https://books.google.com.br/books?id=bLaeAAAAMAAJ
[33] T. de Oliveira, D. Stringhini, and D. G. M. Corrêa, "Online Peer Assessment and Scoring Rubric to Produce Better Digital Systems Designs in an Undergraduate Computer Engineering Curriculum," in *Latin American Conference on Learning Technologies (LACLO)*, 2018.
[34] T. Oliveira, D. Stringhini, J. J. S. Craibas, and D. G. M. Corrêa, "Metodologia de Ensino baseada em Avaliações Colaborativas e Rubricas para o Aprimoramento da Habilidade de Escrita de Relatórios Técnicos em Cursos de Graduação," in *Brazalian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação - SBIE)*, oct 2018, p. 1603. [Online]. Available: http://br-ie.org/pub/index.php/sbie/article/view/8121

Tiago Oliveira possui graduação em Ciência da Computação (2002) e Doutorado em Engenharia Elétrica (2009) pela Universidade Estadual Paulista Júlio de Mesquita Filho. Atualmente é professor da Universidade Federal de São Paulo. Tem experiência na área de Ciência da Computação, com ênfase em Arquitetura de Sistemas de Computação, atuando principalmente nos seguintes temas: síntese de sistemas digitais, arquiteturas reconfiguráveis e linguagens de descrição de hardware. Tem atuado também na área de tecnologia educacional e metodologias de ensino-aprendizagem.

Denise Stringhini concluiu o doutorado em Ciência da Computação em 2002 pela Universidade Federal do Rio Grande do Sul. Possui mestrado em Ciência da Computação pela mesma instituição (UFRGS, 1997) e é graduada em Informática pela PUC-RS (1993). É professora em cursos de Ciência da Computação e Sistemas de Informação desde 1994. Atualmente é professora do Instituto de Ciência e Tecnologia (ICT) da Universidade Federal de São Paulo (UNIFESP) em São José dos Campos (desde 2013). Seus principais interesses de pesquisa se encontram nas áreas de Processamento de Alto Desempenho (PAD) e Sistemas Distribuídos. Tem atuado também na área de tecnologia educacional e metodologias de ensino-aprendizagem.

José J. S. Craibas possui graduação em Ciência e Tecnologia (2016) pela Universidade Federal de São Paulo e Técnico em Mecatrônica (2010) pela Faculdade ENIAC. Atualmente é monitor dos laboratórios de Circuitos Digitais e Arquitetura e Organização de Computadores na Universidade Federal de São Paulo e cursa Engenharia de Computação na mesma instituição. Tem experiência na área de Engenharia de Computação, com ênfase em Arquitetura de Sistemas de Computação e Redes Definidas por Software, atuando principalmente nos seguintes temas: arquiteturas reconfiguráveis, linguagens de descrição de hardware e metodologias de ensino-aprendizagem.