PLANO DE TRABALHO PARA O MESTRADO

DESENVOLVIMENTO DE UM MODELO INSTRUCIONAL PARA O ENSINO DE ENGENHARIA DE *SOFTWARE* E USABILIDADE

Universidade Federal de Santa Catarina – UFSC

Programa de Pós-Graduação em Ciência da Computação – PPGCC

Professora Orientadora: Christiane A. Gresse von Wangenheim

1. Introdução

A computação está cada vez mais presente no nosso cotidiano por meio dos diversos dispositivos digitais que estão se tornando indispensáveis para as nossas tarefas. Dessa forma, é essencial que os profissionais do século XXI, independentemente de sua área de conhecimento, tenham uma compreensão dos princípios e práticas da Ciência da Computação pois ajuda a resolver problemas do mundo real através do uso do pensamento computacional (CSTA,2011). Além disso, no Brasil, existe a necessidade de formar profissionais para o setor de software, pois existe um déficit de profissionais especializado na área (IDC, 2016).

Nesse contexto, diversos trabalhos foram realizados desenvolvendo uma unidade instrucional visando ensinar computação no ensino fundamental. O desenvolvimento de jogos em celulares mostra estar sendo o melhor caminho para o ensino da computação (Daniel et al, 2016). No trabalho de Wagner et. al (2012), a ferramenta App Inventor foi amplamente utilizado para criar apps em *smartphone,* auxiliando no ensino de conceitos da computação, por exemplo a programação. Porém, o guia de referência curricular ACM/CSTA K-12 (2011), utilizado como referência pelos trabalhos para alinhar o ensino da computação no ensino fundamental, define diversas competências da área da ciência da computação que o aluno deve aprender, dentre elas, Engenharia de Software (ES) e Engenharia de Usabilidade (EU).

Visando ensinar o pensamento computacional, ACM/CSTA K-12 (2011) define que os alunos do ensino fundamental 2 devem ser capazes de descrever e aplicar as atividades de um ciclo de vida de engenharia de *software*, por exemplo, *design*, codificação, teste, verificação. SWEBOOK explica que essas atividades tem como benefício: facilitar a compreensão humana, comunicação e coordenação, melhorar a qualidade dos produtos de software de maneira eficiente. USABILIDADE (????). Apesar dessas duas áreas de conhecimento divergirem no foco durante desenvolvimento do software, elas definem modelos de processos, métodos e técnicas para o desenvolvimento de software interativo e sistemático (Gibertoni, 2016).

Atualmente, os trabalhos pesquisados integram superficialmente a ES no ensino fundamental. Christina (artigo 1) e Nathalia (tcc) desenvolveram uma unidade instrucional que tem como um dos objetivos aplicar conhecimentos básicos de ciclo de vida de engenharia de software com a finalidade de projetar resoluções de problemas. No que se refere a EU, a maioria dos estudos pesquisados focam na graduação, como é o exemplo de Sommariva et. al (2012), que desenvolve um jogo chamado UsabilityGames que tem como objetivo facilitar o ensino da EU para os alunos de graduação abordando ciclo de vida de EU, análise de requisitos, avaliação de heurísticas e prototipação. Apesar de existir vários trabalhos para o ensino de ES e EU, mas nenhum integra de forma sistemático a ES e EU no ensino fundamental.

o desenvolvimento sistemático de um modelo de ensino de engenharia de *software* e engenharia de usabilidade para o desenvolvimento de aplicativos móveis no Ensino Fundamental 2 com o ambiente de programação App Inventor

com o objetivo de atender esseas. , como por exemplo, França et al. (2012) e Oliveira (2014), que ensinam os conceitos de programação utilizando a ferramenta Scratch. Porém, nenhum desses trabalhos ensinas conhecimentos das áreas de engenharia de software e usabilidade. Apesar das áreas tem focos diferente, ma

Conforme o guia curricular ACM/CSTA K-12, os alunos do ensino Fundamental 2, visando o ensino do pensamento computacional, devem ter competência em definir atividades relacionado ao processo de desenvolvimento de software, por exemplo, design, codificação, teste, verificação.

SWEBOOK explica que essas atividades tem como benefício: facilitar a compreensão humana, comunicação e coordenação, medir e melhorar a qualidade dos produtos de software de maneira eficiente, entre outros.

Além disso, o guia define que aluno deve ser capaz de avaliar a legibilidade e usabilidade do software.

na perspectiva de habilidade de práticas de computação e programação, devem ser capazes de aplicar métodos de depuração e testes de software, aplicar modelos de ciclo de vida de software, criar e organizar páginas da web através de ferramentas de design de programação web, entre outros.

Atualmente, no Brasil, o ensino da computação limita-se apenas ao ensino superior. A aprendizagem de competências da computação no ensino básico está se tornando cada vez mais necessário. O trabalho realizado por ALVES et al, (2016), desenvolve uma unidade instrucional que aplica os conhecimentos e fundamentos da computação no ensino fundamental de forma multidisciplinar. Foi desenvolvido jogos relacionado à disciplina de História e estudos sociais em dispositivos móveis. O resultado indicou que, além de aprender a disciplina, houve aprendizagem de conceitos básicos da programação e o interesse das crianças nesta área de conhecimento.

A construção de *software* sem a aplicação de técnicas de engenharia de *software* é um desafio pois, a longo prazo, tornará o sistema de baixa confiabilidade, difícil de lidar com as diversidades, manutenção, evolução, e tendo menos aceitabilidade dos usuários (Somerville, 2012). Ainda Somerville 2012, a engenharia de *software* se preocupa desde os estágios iniciais da especificação do sistema até a sua manutenção. Envolve todo o aspecto do desenvolvimento, incluindo atividades de gerenciamento de projetos de *software*, desenvolvimento de ferramentas, métodos e teorias.

Dessa forma, faz-se necessário elaborar um modelo instrucional para aplicar as práticas de engenharia de *software* na inserção da computação no ensino básico. Esse modelo deve seguir as diretrizes do currículo para o ensino da computação CSTA/ACM K-12 (CSTA, 2011) alinhado com o currículo do ensino fundamental 2.

Atualmente, existem diversas ferramentas para auxiliar a aprendizagem da programação, como por exemplo, a ferramenta App Inventor(APP INVENTOR, 2014). Esta ferramenta permite construir aplicativos para celulares de forma fácil e pedagógica. Contudo, percebe-se que é necessário customizar alguns componentes de telas da ferramenta, pelo fato dos mesmos não terem uma interface visual amigável. A aplicação de princípios da disciplina de engenharia de usabilidade facilitará na criação de aplicativos móveis mais atrativos. Conforme Cybis et al, (2010), o desafio da engenharia de usabilidade é desenvolver sistemas analisando cuidadosamente os diversos componentes de seu contexto de uso para atender as expectativas de interação com uma interface simples, intuitiva e fácil de usar.

1. Objetivos

Este trabalho tem como objetivo o desenvolvimento sistemático de um modelo de ensino de engenharia de *software* e engenharia de usabilidade para o desenvolvimento de aplicativos móveis no Ensino Fundamental 2 com o ambiente de programação App Inventor. O desenvolvimento deste modelo engloba a definição de um processo de desenvolvimento de apps integrando sistematicamente práticas de Engenharia de Software e Engenharia de Usabilidade.

O objetivo desse trabalho está inserido na linha de pesquisa de Engenharia de *Software* do PPGCC dentro dos tópicos de Processo de Desenvolvimento de *Software* conforme a definição da área de engenharia de *software* da SBC (SBC,2005) e alinhado a norma ISSO/IEC 12207.

Objetivos específicos:

O1. Analisar a fundamentação teórica sintetizando os conceitos básicos em relação ao ensino de computação no ensino fundamental 2, ao ambiente de programação App Inventor, e à engenharia de *software* e usabilidade

O2. Levantamento do estado da arte e prática por meio da revisão sistemática de literatura para entender como atualmente os conceitos de engenharia de *software* e de usabilidade são ensinados no nível do ensino fundamental.

O3. Evoluir uma unidade instrucional para o ensino de desenvolvimento de aplicativos integrando o ensino de engenharia de *software* e usabilidade.

O3.1. Definir um processo de engenharia de *software* e usabilidade voltado ao desenvolvimento de aplicativos no contexto do ensino fundamental 2.

O3.2. Desenvolver material didático, como por exemplo, slides, roteiros, folhas de tarefas, avaliações.

O3.3. Adaptação/evolução do App Inventor para apoiar o ensino do processo de desenvolvimento de *software*.

05. Aplicar e avaliar a unidade instrucional desenvolvida em escolas avaliando a unidade em relação à aprendizagem dos alunos, bem como sua efetividade.

3. Contribuições científicas potenciais.

Este trabalho tem como principal contribuição científica a elaboração de um modelo de ensino de engenharia de *software* e usabilidade para o ensino fundamental 2. Além disso, prevê-se as seguintes contribuições científicas:

- Levantamento do estado de arte e prática de forma sistemática fornecendo uma visão geral sobre esta questão de pesquisa;

- Definição de um processo de desenvolvimento de apps integrando práticas de engenharia de software e usabilidade no contexto do ensino fundamental;

- Desenvolvimento de uma unidade instrucional para introduzir práticas de engenharia de *software* e usabilidade no ensino de computação no nível de ensino fundamental 2.

- Dados e resultados de avaliação sistemática do ensino de engenharia de *software* e usabilidade por meio de estudo de caso;

- Aprimoramento do ambiente de programação App Inventor para suportar adequadamente o processo definido e o modelo de ensino.

- Como impacto social o presente projeto visa a popularização da computação contribuindo de forma prática a sua aplicação contribuindo para o crescimento dessa área de conhecimento.

Referências

CSTA. ACM. **CSTA K –12 Computer Science Standards**, 2011. Disponível em: < http://c.ymcdn.com/sites/www.csteachers.org/resource/resmgr/Docs/Standards/CSTA\_K-12\_CSS.pdf>. Acesso em: Abril 2017.

Franc¸a, R. S., Silva, W. C., and Amaral, C. J. H. (2012). Ensino de ciencia da computac¸ ˆ ao˜ na educac¸ao b ˜ asica: Experi ´ encias, desafios e possibilidades. In ˆ Anais do XXXII Congresso da Sociedade Brasileira de Computac¸ao

IDC, 2016. <http://www.cisco.com/assets/csr/pdf/IDC_Skills_Gap_-_LatAm.pdf>

Oliveira, M. L. S., de Souza, A. A., Barbosa, A. F., Barreiros, E. F. S. (2014) “Ensino de lógica de programação no ensino fundamental utilizando o Scratch: um relato de experiência”. In: In: Anais do Workshop de Informática na Escola, Brasília – DF.

Wangenheim, C. et al. **Ensinando a programação por meio de programção com App Inventor.** 2016.

ALVES, N. et al. **Ensino de Computação de Forma Multidisciplinar em Disciplinas de História no Ensino Fundamental – Um estudo de Caso.** Artigo submetido para Revista Brasileira de Informática na Educação. v. 24, n. 3, Mar. 2016.

APP INVENTOR, **About us**. Disponível em: <http://appinventor.mit.edu/explore/about-us.html>. Acesso em: 20 de abril de 2017.

CSTA. ACM. **CSTA K –12 Computer Science Standards**, 2011. Disponível em: < http://c.ymcdn.com/sites/www.csteachers.org/resource/resmgr/Docs/Standards/CSTA\_K-12\_CSS.pdf>. Acesso em: Abril 2017.

G Morati Jr, Roberto, et al. (2012) "Alice no Labirinto das Decisões: um jogo para

exercitar a tomada de decisão e planejamento." *Anais dos Workshops do Congresso*

*Brasileiro de Informática na Educação*. Vol. 1. No. 1.

Cybis, W., Betiol, A. H. e Faust, R. **Ergonomia e usabilidade: conhecimentos, métodos e aplicações**. Editora Novatec, 2ª edição, 2010.

Rozelma 2012, **Ensino de Ciência da Computação na Educação Básica: Experiências, Desafios e Possibilidades (artigo)**

SBC. **Currículo de Referência da SBC para Cursos de Graduação em Bacharelado em Ciência da Computação e Engenharia de Computação**. Sociedade Brasileira de Computação, 2005.

SOFTEX. **Relatório anual 2013**. Associação para promoção da Excelência do *Software* Brasileiro. Campinas/SP, p. 39. 2013.

SOMMERVILLE, I. **Engenharia de *Software***. [S.l.] : Editora Pearson Education, nona edição. São Paulo, 2012.

3-A

Pensamento Computacional

1. Descrever um processo de desenvolvimento de software Usado para resolver problemas de software (por exemplo, design,Codificação, teste, verificação).

Colaboração

1. Trabalhar em equipe para projetar e desenvolver Software.

Prática de Computação e Programação (CPP)

* 1. Criar e organizar páginas da Web através do uso de uma variedade de ferramentas de design de programação web. (EU)
* 3. Usar vários métodos de depuração e teste Para assegurar a correção do programa (por exemplo, teste Casos, teste de unidade, caixa branca, caixa preta, teste de integração)
* 4. Aplicar análise, desenho e técnicas de implementação para resolver problemas (por exemplo, usar um ou mais modelos de ciclo de vida de software).

3 B

Colaboração

3. avaliar os programas escritos por outros para a legibilidade e usabilidade.