Scratch na produção de recursos interdisciplinares com disciplinas indígenas

Hévellyn de Morais Rabêlo, Wênnio Kelson F. de Oliveira, Luan de Luna Santos, Ana Liz Souto O. de Araújo, Flávia Veloso C. Souza

Centro de Ciências Aplicadas e Educação – Universidade Federal da Paraíba (UFPB)
Rua da Mangueira, s/n – CEP 58.297-000 – Rio Tinto – PB – Brasil

{hevellyn.morais, wennio.kelson, luan.luna,analiz,flavia}@dcx.ufpb.br

Abstract. *Integrate computation at primary education is a current challenge. This work is an initiative that provides planning, implementation and evaluation of a course that integrate computational thinking by programming in Art and Culture and Tupi Language disciplines at Indian schools. Scratch was used to produce animations in the contents of indigenous culture. The animations were evaluated by a framework to measure the computational thinking with Scratch code.*

Resumo. *Integrar o ensino de computação as atividades do ensino básico é um desafio atual. Nessa vertente, este trabalho é uma iniciativa que apresenta o planejamento, a execução e a avaliação de um curso que buscou integrar o pensamento computacional por meio da programação nas disciplinas de Arte e Cultura e Língua Tupi em escolas indígenas. Foi utilizado o Scratch como ferramenta para produção de animações abordando conteúdos da cultura indígena. As produções dos alunos foram avaliadas segundo um framework para mensurar habilidades do pensamento computacional exploradas por meio de código Scratch.*

1. Introdução

O uso da informática como ferramenta auxiliar no ensino e aprendizagem das disciplinas do currículo escolar brasileiro é de extrema importância e é tema de discussões na área. A carência de iniciativas que trabalhem com os alunos a interdisciplinaridade envolvendo a computação e as disciplinas do currículo escolar brasileiro ainda é grande.

Alguns autores afirmam a importância que os alunos aprendam cada vez mais cedo a desenvolver habilidades do pensamento computacional. Embora o termo "pensamento computacional" careça de definição e maiores estudos, ele é definido pela CSTA (*Computer Science Teachers Association*) como uma forma de pensar e resolver problemas utilizando conceitos da ciência da computação. Trata-se de um conjunto de conceitos, habilidades, práticas e ferramentas da computação que podem ser aplicadas tanto no cotidiano como em diversas áreas do conhecimento [Wing, 2006]. Mas então, como ou o que deve ser feito para auxiliar os alunos de forma que esses desenvolvam tais habilidades?

Neste contexto, surgem iniciativas de integrar o desenvolvimento do pensamento computacional com os conteúdos do ensino básico usando ferramentas computacionais. Uma das ferramentas que podem ser utilizadas nesse meio é o *Scratch*, como relata [França *et al*, 2013]. Ademais, outros autores corroboram que o ensino e aprendizagem de conceitos introdutórios de programação por meio da linguagem *Scratch* são fatores de sucesso [Araújo *et al*, 2013] e [Pereira *et al*, 2012]. Experiências relatam o desenvolvimento de criatividade, raciocínio lógico, pensamento abstrato, por meio dessa ferramenta.

Diante desse cenário, os licenciandos em Ciência da Computação da UFPB (Universidade Federal da Paraíba) no município de Rio Tinto/PB vislumbraram uma oportunidade de integrar computação com a cultura local. Essa região tem como particularidade o fato de ter um grande número de habitantes índios ou descendentes de índios, e de possuir escolas públicas destinadas exclusivamente a eles. Elas buscam resgatar e perpetuar a cultura indígena por meio de disciplinas específicas, como Arte e Cultura e de Língua Tupi antiga. Elas fazem parte da grade curricular obrigatória e do projeto político pedagógico das escolas da região, devido à singularidade da localidade.

Assim, foi percebida a oportunidade de se planejar um curso utilizando a linguagem de programação *Scratch* para conceber atividades interdisciplinares, visando desenvolver o pensamento computacional integrado as disciplinas do currículo básico local de Arte e Cultura e Língua Tupi antiga. As atividades visam estimular o pensamento computacional nos alunos, por meio da prática de habilidades relacionadas à programação com *Scratch*, que serão transmitidos fazendo paralelos com a cultura dos discentes.

Apesar de existirem outras abordagens utilizando a computação junto a outras disciplinas, neste trabalho apresenta uma peculiaridade que se dá pelo seu contexto sociocultural indígena. As disciplinas foram selecionadas pela pesquisa com o objetivo de valorizar, realçar e fortalecer a importância da cultura, arte e língua para os índios e descendentes.

O artigo está organizado da seguinte maneira: a seção 2 apresenta uma breve explanação sobre o conceito e habilidades do pensamento computacional adotadas neste trabalho; a seção 3 cita características do *Scratch*; a seção 4 aborda o planejamento metodológico e relata a aplicação o curso; a seção 5 traz os resultados e discussões; a seção 6 apresenta as considerações finais e trabalhos futuros.

2. Pensamento computacional

A *Computer Science Teachers Association* (CSTA) reconheceu a relevância do pensamento computacional na educação básica e criou a *Computational Thinking Task Force* com o objetivo de explorar, disseminar e recomendar recursos para ensino e aprendizagem relacionados ao pensamento computacional. Ela conta com a parceria da *Internacional Society for Technology in Education* (ISTE) para desenvolver definições operacionais do pensamento computacional e oferecer ferramentas e recursos para os professores utilizarem em todos os níveis de ensino e em áreas de estudo.

A CSTA define o pensamento computacional como um processo de resolução de problemas nos quais estão relacionados habilidades de: Formular problemas de forma a

se beneficiar de conceitos, de ferramentas e de recursos computacionais para ajudar a resolvê-los; organizar e analisar dados; representar dados através de abstrações, identificando modelos e realizando simulações; automatizar soluções através do pensamento algorítmico; identificar, analisar e implementar soluções com o objetivo de alcançar a combinação mais eficiente e eficaz de passos e recursos; generalizar e transferir o processo de solução de um problema para uma gama de problemas semelhantes.

Essas habilidades são apoiadas e reforçadas por um conjunto de atitudes essenciais a prática do pensamento computacional. Tais atitudes estão relacionadas à capacidade de: lidar com problemas que tenham diferentes níveis de complexidade; ter persistência ao trabalhar com problemas difíceis; lidar com problemas ainda não resolvidos; e se comunicar e trabalhar em grupo para atingir um objetivo em comum.

Um exemplo de habilidades do pensamento computacional relacionadas diretamente com o Scratch são recomendadas por [Resnick *et al*, 2011]. Os autores listaram os conceitos computacionais: sequência; loop; paralelismo; evento; condicionais; operadores e dados. Bem como as práticas computacionais: iterativo e incremental; testar e fazer uso de debug; reusar e recombinar e abstrair e modularizar. Segundo os autores, são essas as habilidades relacionadas com programação esperadas quando se desenvolvem atividades que buscam explorar o pensamento computacional com *Scratch*.

3. Scratch

O *Scratch* foi criado pelo *Massachusetts Institute of Technology* (MIT), no ano de 2003, e se trata de um software educacional que tem como propósito ensinar programação [Lifelong Kindergarten, 2007]. Ele tem em sua interface um ambiente de desenvolvimento atrelado a uma linguagem de programação, permitindo que o usuário, ao mesmo tempo em que acesse a sintaxe desta linguagem, construa seu código.

O *Scratch* é uma ferramenta que permite ao usuário uma experiência de programação visual, propiciando explorar práticas, perspectivas e conceitos computacionais já citados na seção anterior, e que são fundamentais na dispersão do pensamento computacional. Segundo [Brennan, 2011] tais práticas, perspectivas e conceitos são trabalhados no *Scratch* de forma criativa onde a aprendizagem está baseada no conceito de design e que, por sua vez, enfoca quatro processos distintos, sendo eles: o processo de concepção, ou seja, a arte de produzir ao invés de simplesmente utilizar de maneira interativa; a personalização que permite que as produções sejam relevantes e significativas ao produtor; a reflexão que interpreta e repensa as atividades dos colegas e por fim a colaboração relacionada ao processo de desenvolver atividades em grupo.

O *Scratch* disponibiliza ao usuário o despertar de habilidades importantes para o processo de aprendizagem. Um exemplo dessas habilidades é a aquisição de dados, adquiridos ao usar os diversos tipos de mídia como imagens, textos e áudios, podendo esses serem importados ou gravados. Ao trabalharem com os conceitos computacionais via *Scratch*, os usuários estão aprendendo a programar, e dessa forma vão aprimorando o pensamento sistemático, o raciocínio lógico e demais habilidades que, ao se somarem, lhes trarão uma melhor compreensão e prática do pensamento computacional.

4. Planejamento metodológico e aplicação do curso

O curso teve seu planejamento orientado pela abordagem construtivista, a qual afirma que o conhecimento pode acontecer por meio de ideias que já foram construídas anteriormente pelos indivíduos, partindo do meio no qual está inserido. [Costa, 2009] escreve sobre a necessidade de compreender que os estudantes levam ao convívio escolar, uma série de habilidades, hábitos e costumes, por meio dos seus valores, da sua cultura, de suas crenças, de suas atitudes e dos significados que trazem consigo. Assim, jugou-se que tal abordagem seria adequada ao contexto sociocultural da escola e dos alunos indígenas.

O objetivo principal do curso foi trabalhar habilidades do pensamento computacional, por meio da programação, utilizando a cultura local dos alunos indígenas. Como ferramenta, foi escolhida a plataforma *Scratch* pela sua facilidade de uso nas construções de algoritmos e animações.

As disciplinas selecionadas para realizar a atividade interdisciplinar foram Língua Tupi e Arte e Cultura, nas turmas de 1° ano do ensino médio e 9° ano do ensino fundamental, respectivamente. Ambas as disciplinas trabalham o vocabulário da língua Tupi antiga, bem como costumes e rituais indígenas, falados e vividos pelas tribos locais. Tais disciplinas compõem o currículo obrigatório das duas escolas: Escola Indígena de Ensino Fundamental e Médio Guilherme da Silveira e Escola Indígena de Ensino Fundamental e Médio Doutor José Lopes Ribeiro, ambas no município de Rio Tinto/PB. Os alunos de ambas as escolas são das aldeias Jaraguar e Monte Mor, as quais estão localizadas nas proximidades das escolas citadas.

O curso foi planejado com carga horária de 16h, dividido em quatro aulas (quatro módulos) e realizado nos laboratórios da UFPB, Campus IV, em Rio Tinto. Participaram do curso alunos do 9° ano do ensino fundamental e 1° ano do ensino médio das escolas citadas, e um professor de informática de uma terceira escola indígena da região, sendo um total de 27 alunos, 12 do sexo feminino e 15 do sexo masculino.

A primeira aula abordou teorias que embasam os conteúdos introdutórios de programação. Assim, foi explicado o conceito de programação e a sua relação com os conteúdos que seriam abordados no curso e com a ferramenta *Scracth*. Nesse momento, foram feitas analogias utilizando a rotina dos discentes e os assuntos do curso, buscando facilitar a compreensão deles com os primeiros conceitos computacionais abordados. Por exemplo, para explicar o conceito de sequência, foi construído um algoritmo que relatava a rotina do caminho da aldeia até a escola. Para isso, os alunos organizaram o passo a passo desde a saída da aldeia até a chegada na escola. Dessa forma, o algoritmo foi construído permitindo a discussão da ideia de passos e sequência, bem como do conceito de colaboração, tendo em vista que todos os alunos auxiliaram na construção do algoritmo.

A segunda aula teve como ponto de partida a revisão dos conteúdos vistos na primeira aula versando teoria e prática. Depois, foram abordados conceitos e práticas utilizando laço de repetição e variáveis no *Scratch*. Nesta aula também focou-se no conceito de evento e foram propostas práticas empregando os blocos de comando movimento, controle e aparência, presentes na ferramenta.

Na terceira aula, o conceito de evento foi reforçado, dando ênfase ao paralelismo de eventos. Foi proposta uma atividade com objetivo de trabalhar criatividade e eventos. Essa atividade propôs a construção de uma animação de temática livre no *Scratch*. Os alunos foram instigados a pensarem no que eles desejavam construir e depois como eles expressariam isso no *Scratch*. Foi dado orientações que eles poderiam reusar trechos de códigos de atividades anteriores. Nesse módulo, também foi possível trabalhar as práticas computacionais de teste e correção de erros, pois, os alunos tiveram auxilio dos monitores sobre como analisar o código desenvolvido no *Scratch* para identificar seus bugs, corrigindo-os e testando-os até que o seu código estivesse correto.

A quarta aula foi iniciada com uma revisão dos conteúdos ministrados durante o curso. Neste módulo, os alunos foram estimulados a desenvolver o projeto final do curso: uma animação que explicasse um ritual da tribo ou palavras na língua Tupi já trabalhadas na sala de aula pelo professor da disciplina correspondente. Tratou-se de um exercício interdisciplinar que buscou, mais uma vez, estimular o pensamento computacional e o conteúdo das disciplinas locais. A figura 1 mostra um exemplo de uma animação construída por um aluno do curso.

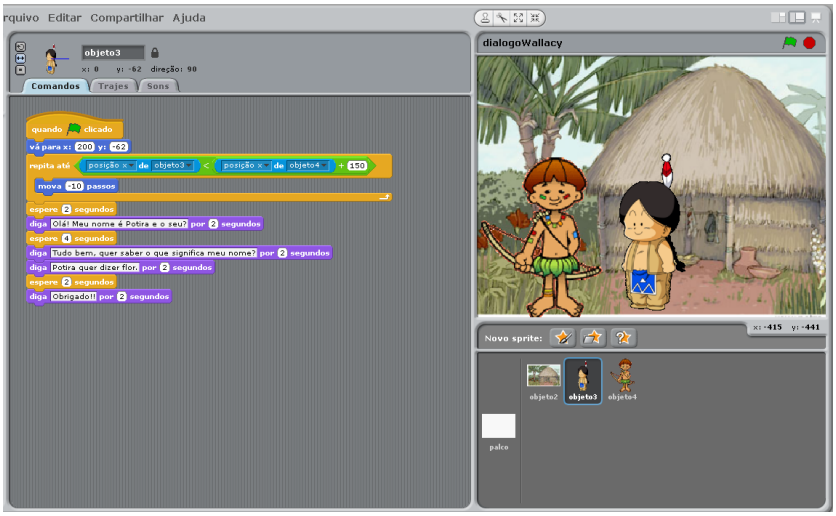

Figura 1 - Exemplo de animação produzida no projeto final

5. Resultados e discussões

Após a produção das animações das aulas 3 e 4, buscou-se uma forma de quantificar e levar indícios se as práticas realizadas com *Scratch* permitiram trabalhar habilidades do pensamento computacional, e em qual nível de profundidade. Assim, pesquisou-se na literatura formas de realizar uma avaliação inicial sobre os recursos produzidos pelos alunos no *Scratch* relacionados ao pensamento computacional.

Devido as poucas propostas na literatura de se avaliar objetivamente o desenvolvimento de habilidades do pensamento computacional por meio de programação com *Scratch*, escolheu-se usar o *framework Progressions of Early Computational Thinking* (PECT) [Seiter *et al*, 2013]. O PECT foi o escolhido, pois consiste em um modelo que mensura as progressões do pensamento computacional via *Scratch*, por meio de uma sintetização que quantifica os códigos desenvolvidos pelos alunos em três níveis: básico, intermediário e profissional. O modelo funciona da

seguinte forma: observa-se a utilização de padrões de projeto na codificação dos alunos e realiza-se classificações. Em outras palavras, à medida que os discentes fazem uso de elementos de programação que são comumente utilizados através de padrões de projeto, se torna possível fazer o mapeamento dos conceitos do pensamento computacional que estão sendo aplicados nos recursos produzidos.

Como o PECT é um modelo abrangente, foi utilizado apenas o primeiro subconjunto do *framework* para avaliar as atividades dos alunos. Tal decisão foi tomada porque esse primeiro subconjunto é suficiente para classificar o conteúdo ministrado no curso. O Quadro 1 apresenta a adaptação do PECT utilizada neste trabalho. Na primeira coluna observa-se os conceitos computacionais; na primeira linha, os níveis de conhecimento adquirido em cada conceito; as intersecções são os conjuntos de comandos do *Scratch* que, segundo o modelo, mensuram em que nível de conhecimento o aluno chegou utilizando o referido comando.

Quadro 1 - Modelo de avaliação adaptado de Seiter et al [2013]

	Básico	**Intermediário**	**Profissional**
Looks (bloco aparência)	Diga, pense.	Próxima fantasia, mostre, esconda.	Mudar para fantasia, mudar a cor, tamanho etc.
Sons	Reproduza o som, a nota.	Tocar X tocar até fazer.	
Movimento	Mover, vá para o Sprite/ponto, vire.	Ir para x, y. Deslizar para x, y.	*Set*, a mudança X, Y.
Sequencia e loop (bloco controle)	Sequencia.	Repita, sempre.	Sempre se, repita até.
Expressões booleanas (bloco operadores)	Operadores de detecção.	<, =, >.	*And, or, not.*
Condicionais (bloco de controle)	*If.*	*If...else.*	Junção do *if* com *if...else.*
Coordenadas (bloco de movimento)	Espere.	Quando eu receber, *broadcast.*	Espere ate.
Interface de usuário de evento (bloco de controle)	Quando bandeira verde clicado.	Quando chave pressionada, quando objeto clicado.	Anuncie para todos e espere.
Paralelismo (bloco de controle)		Dois scripts iniciam com o mesmo evento.	
Inicializar localização (bloco de controle/movimento)		Quando bandeira verde clicado, Defina propriedades de localização (x, y, etc.).	
Inicializar looks (bloco de controle/aparência)		Quando bandeira verde clicada mude as propriedades dos looks (traje, visibilidade, etc.).	

Para cada projeto realizado pelos alunos nas aulas 3 e 4, foi gerada uma tabela avaliativa correspondente, que realizou a classificação de acordo com o modelo proposto. Após essa etapa, os resultados individuais das tabelas foram condensados em uma tabela geral por projeto. A partir dessas tabelas de cada projeto, foram concebidos os gráficos que serão exibidos em seguida.

A Figura 3 apresenta o gráfico de avaliação do desempenho da turma durante o projeto da terceira aula. O eixo horizontal do gráfico apresenta a quantidade de alunos que atingiram cada variável de projeto. Já o eixo vertical representa os conceitos computacionais do modelo PECT, enquanto que as cores representam o nível básico (azul), intermediário (vermelho) e profissional (verde). Ao fim de cada barra de nível é apresentado o valor exato de alunos que atingiram o nível em questão. Essa atividade foi realizada por 17 alunos.

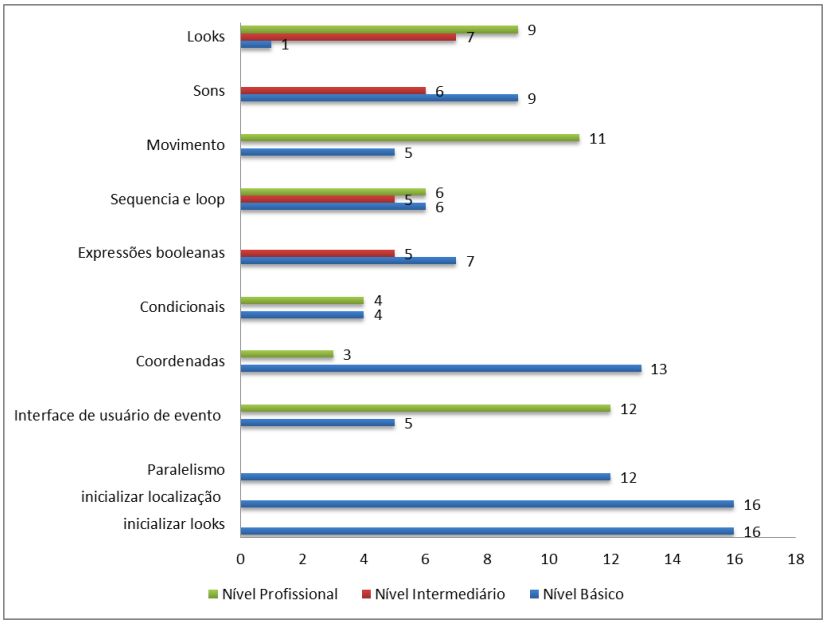

Figura 2 - Gráfico de avaliação do projeto da terceira aula

Na Figura 3 é apresentado o gráfico de avaliação do desempenho da turma no projeto final. Não foi possível avaliar o quesito Som nesta atividade devido a problemas técnicos nos computadores durante a aula (microfone não identificado pela placa de som no momento da prática). Este gráfico segue o mesmo esquema do gráfico anterior. Essa atividade foi realizada por 15 alunos.

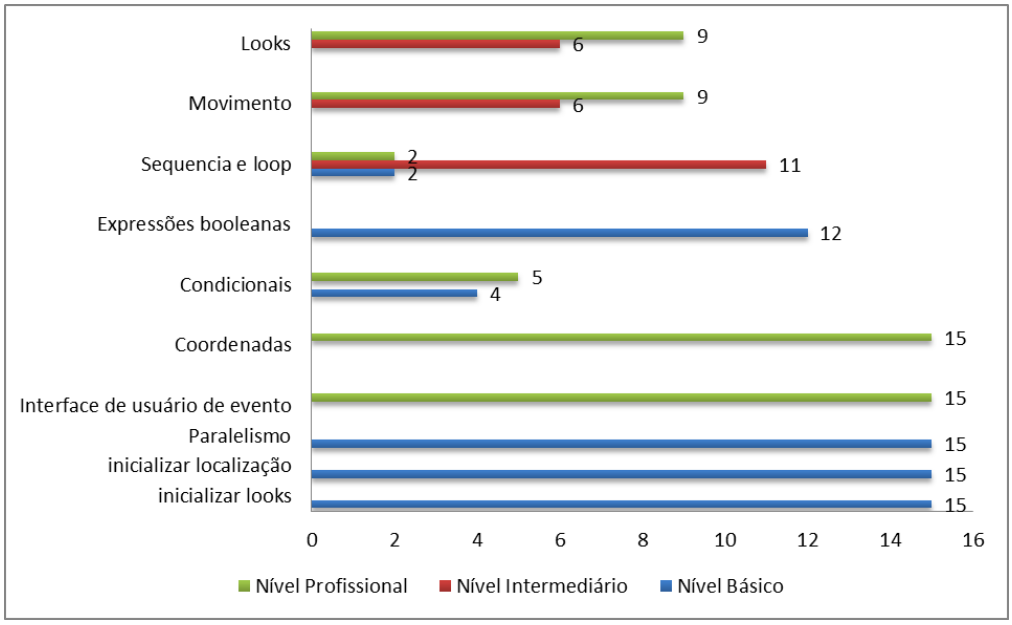

Figura 3 - Gráfico referente ao projeto da quarta aula

Os três últimos critérios do modelo (paralelismo, iniciar localização e iniciar looks) são verificados apenas como presentes ou ausentes nos códigos desenvolvidos pelos alunos. Assim, tais variáveis são padronizadas como nível básico se estiverem presentes nos códigos.

Observando os gráficos, vê-se que os comandos básicos de *Looks*, Movimento e Sequência, os quais estão diretamente relacionados com a construção de algoritmos, foram usados por mais da metade dos alunos em ambas as atividades. Já as expressões condicionais foram pouco empregadas nas animações criadas.

Duas das habilidades do pensamento computacional bastante frisada no curso foram os conceitos de paralelismo e sincronismo. Tais conceitos eram fundamentais para construção da sequência correta dos diálogos, nos quais os tempos de início e término da fala de cada personagem precisariam ser harmonizadas e sequenciadas. Nos dois projetos, apenas um aluno não soube utilizar blocos de controle.

Um destaque sobre o paralelismo e sincronismo foi observado. Ocorreu que, do início para o meio da animação realizada no projeto final, o sincronismo foi usado corretamente, respeitando o tempo de diálogo entre os personagens. Entretanto, do meio para o final, o sincronismo não obteve a mesma qualidade do início. Pode-se justificar esse acontecimento pelo fato do tempo destinado ao projeto final não ter sido suficiente para os alunos desenvolverem a atividade por completo com qualidade. Apesar disso, acredita-se que os alunos compreenderam e sabem utilizar o sincronismo, pois eles utilizam corretamente tanto na primeira parte da exercício, como na aula anterior.

Além dos resultados apontados pela aplicação do PECT, durante o curso, observou-se o interesse dos alunos em relação ao desejo de utilizar o *Scratch* e produzir animações. Percebeu-se que os recursos visuais e multimídia são atrativos e conseguem capturar a atenção para atingir um objetivo proposto, além de ser um recurso que facilita o desenvolvimento inicial de habilidades do pensamento computacional, como abstração, sequência, paralelismo. Observou-se também que, mesmo para um curso de

curta duração, o fato dos blocos de comando do *Scratch* serem agrupados por cor, segundo os alunos, facilitou a localização individual de comandos.

O discente e também professor de informática, já mencionando anteriormente, afirmou que o curso foi valioso para sua carreira como docente, para despertá-lo para novas estratégias de ensino de conceitos de computação. Ele afirmou que pretende propagar os ensinamentos em suas aulas, inclusive desenvolvendo atividades com *Scratch* para alunos de séries iniciais do Ensino Fundamental, tendo em vista que a ferramenta é de fácil manuseio e compreensão. Ele se mostrou favorável a trabalhar de forma interdisciplinar com outras disciplinas do currículo básico e não apenas limitado suas aulas ao uso de ferramentas de Office e internet.

6. Conclusões

Estimular o pensamento computacional através de atividades interdisciplinares integrando programação, arte e cultura e língua Tupi foi o objetivo desse trabalho. Essa meta foi alcançada por meio do desenvolvimento das habilidades do pensamento computacional e os conceitos introdutórios de programação ministrados nesse curso utilizando *Scratch* como ferramenta.

Escolheu-se as disciplinas de Língua Tupi e Arte e Cultura pela possibilidade de criar interdisciplinaridade de computação com a cultura da cidade natal onde o curso foi aplicado. Além disso, vislumbrou-se o potencial que essas disciplinas possuíam para criação de histórias e animações, nas quais eram possíveis desenvolver habilidades do pensamento computacional. Dessa forma, os alunos, que já tinham o conhecimento de sua cultura e vocabulário exercitado nas disciplinas, puderam aprender sobre conceitos computacionais e usá-los para construir e expressar seus costumes e língua.

A avaliação deste trabalho contemplou as atividades desenvolvidas pelos alunos bem como a percepção dos licenciandos que ministraram o curso. Foram planejanejadas atividades que instigavam a criatividade, bem como as habilidades do pensamento computacional, expressadas e aplicadas pelos conceitos computacionais (paralelismo, sequência, operadores, *loops*, eventos, condições) presentes nas estruturas/comandos da programação com *Scratch*. Ademais, a concepção dessas atividades retratam também práticas computacionais (como reuso, abstração, debug, teste, concepção incremental) e outras perspectivas computacionais, tendo em vista que os estudantes se questionaram sobre o que poderiam conceber e como expressariam que desejavam.

Referências

Araújo, A. L. S. O; Scaico, P. D; Paiva, L. F; Rabêlo, H. de M; Santos, L. de L; Pessoa, F. I. R; Targino, J. M; Costa, L. dos S. (2013) Aplicação da Taxonomia de Bloom no ensino de programação com *Scratch*. In: XIX Workshop de Informática na Escola, Campinas – São Paulo.

Brennan, K. (2011). "Creative computing: A design-based introduction to computational thinking". Disponível em: http://scratched.media.mit.edu/sites/default/files/CurriculumGuide-v20110923.pdf. Acessado em: jan. 2015.

Costa, T. A importância da gestão democrática no fortalecimento da qualidade de ensino. (2009). 39 f. Dissertação – Universidade Candido Mendes. Rio de Janeiro - 2009.

Lifelong Kindergarten (2007). Disponível em: http://llk.media.mit.edu/projects.php?id=783. Acessado em: 07 dez 2014.

Pereira, P. de S.; Medeiros, M.; Menezes, J. W. M. (2012) Análise do *Scratch* como Ferramenta de Auxílio ao Ensino de Programação de Computadores. In: XL Congresso Brasileiro de Educação em Engenharia, Belém - Pará.

Resnick, M; Maloney, J; Monroy-Hern ´Andez, A; Rusk, N; Eastmond, E; Brennan, K;Millner, A; Rosenbaum, E; Silver, J; Silverman, B; Kafai, Y. 2011. "Scratch: Programming for all". Comm. ACM 52, 11, 60–67.

Seiter, L; Foreman, B. Modeling the learning progressions of computational thinking of primary grade students. In: Proceedings of the ninth annual international ACM conference on International computing education research. ACM, 2013. p. 59-66. APA

Wing, J. (2008) Computational thinking and thinking about computing. In: Philosophical Transactions of the Royal Society, 2008. p. 3717–3725.