Framework de ensino de programação para crianças e jovens por meio de aprendizado baseado em projetos usando computação tangível, *storytelling*, internet das coisas e sistemas embarcados

Kaique L. Leite[1]**,Kalinka R. L. J. C. Branco**[1]

[1]Instituto de Ciências Matemáticas e de Computação – Universidade de São Paulo (USP)
Caixa Postal 668 – 13.560-970 – São Carlos – SP – Brasil

kaique.leite@usp.br, kalinka@icmc.usp.br

Abstract. *This study describes a low-cost project-based learning framework to teach programing (FEP) and to be used in classrooms. The framework consists of concepts of tangible computing, internet of things and embedded systems. The goal is to teach programming, logic and to practice school's subjects in a fun, interactive, practical and collaborative way, in which the child can feel part of a great and important project for the story being told.*

Resumo. *Esse estudo descreve um framework de ensino de programação (FEP) baseado em projetos e de baixo custo para ser usado em salas de aula. O framework é composto por conceitos de computação tangível, internet das coisas e sistemas embarcados. A meta é ensinar programação, lógica e fixar assuntos vistos em sala de aula de um modo divertido, interativo, prático e colaborativo, no qual a criança possa se sentir parte de um projeto grande e de importância para a história sendo contada.*

1. Introdução

Nas últimas décadas, segundo [Gregio 2004], o desenvolvimento de novas tecnologias nas áreas de comunicação e informação está transformando de modo profundo nossa sociedade. Hoje, os computadores e suas tecnologias fazem parte do dia-a-dia de incontáveis segmentos da sociedade, o que cria novas exigências e competências no âmbito educacional. Para suprir as necessidades educacionais de tais mudanças, [Gregio 2004] salienta que se faz necessária a criação de metodologias inovadoras e mudanças curriculares que satisfaçam as novas necessidades de uma sociedade globalizada. [Fidalgo Neto et al. 2009] acrescenta que, normalmente, o ensino relacionado a novas tecnologias enfatiza o computador, isolado das competências realmente necessárias para o domínio dos novos paradigmas da era digital.

Por esses motivos, primeiro será explorada a literatura relacionada que é dividida em cinco conceitos essenciais: linguagens de programação tradicionais, computação tangível, sistemas embarcados, *storytelling* e ensino com base em projetos (ou do inglês *Project Based Learning* (PBL)).

Segundo [Radoslava et al. 2016] linguagens de programação tradicionais não representam um modo fácil e efetivo de ensinar crianças entre 8 e 9 anos. Nessa idade,

DOI: 10.5753/cbie.wcbie.2017.1301

as crianças ficam amedrontadas com linguagens tradicionais, porque o pensamento abstrato só começa a se manifestar após as idades de 12 e 13 anos. [Bers and Horn 2010] argumentam que as linguagens de programação tendem a ser baseadas em texto, o que torna a experiência das crianças não intuitiva e de certo modo frustrante já que elas ainda estão aprendendo a ler. Mesmo linguagens visuais que funcionam por meio de arrastar, soltar e conectar blocos são um problema para crianças mais jovens pois elas precisam usar interfaces que necessitam de *mouse* e movimentos finos de coordenação, algo que essas crianças ainda estão desenvolvendo. Esse problema faz com que elas tenham que estar acompanhadas de adultos o que, em uma sala com uma quantidade significativa de crianças, é uma tarefa muito difícil.

Já linguagens tangível, por outro lado, segundo [Bers and Horn 2010], utilizam aspectos físicos dos objetos. Sendo assim, o usuário deve conectar e organizar diferentes objetos em uma ordem lógica para construir programas. Os autores afirmam que ao invés do usuário ter que aprender regras e convenções de uma linguagem de programação, os objetos usados na programação tangível oferecem regras naturais de sintaxe através de tamanho, forma e material. Em [Smith et al. 2011] e [Bers et al. 2010] os autores acrescentam que a utilização de blocos físicos é uma técnica eficaz pois oferece acessibilidade ao utilizar objetos conhecidos à infância, ao mesmo tempo em que desenvolve as habilidades necessárias de programação. Com isso, até mesmo as crianças mais jovens podem ter acesso aos conceitos de computação. Além disso, [Horn et al. 2009] em seus estudos afirmam que interfaces tangíveis podem oferecer vantagens significativas em relação a interfaces gráficas usando um *mouse*. Interfaces tangíveis podem ser mais convidativas, serem significativamente mais propícias a participação ativa em grupo do usuário e levar a um tempo maior de interação.

No caso dos sistemas embarcados, [Fagin et al. 2001] afirmam que a utilização de robôs torna muito mais fácil a explicação de conceitos de programação pois esses são inerentes e naturais às ações de um robô. Segundo os autores, essa abordagem possibilita o ensino de conceitos como controle sequencial, variáveis, constantes, procedimentos, expressões booleanas e vetores (*arrays*). Em [Bers and Horn 2010] os autores defendem que a robótica/sistemas embarcados oferecem várias oportunidades para crianças aprenderem sobre sensores, motores, conceitos digitais, matemática aplicada, método científico de investigação, resolução de problemas entre outros. Com isso, crianças são instigadas a participar de interações sociais e negociações enquanto brincam e constroem projetos. A utilização de sistemas embarcados interligados, móveis e que permitem o armazenamento e recuperação dessas informações em nuvem, permite a utilização do que se conhece por Internet das Coisas (ou do inglês *Internet of Things* - IoT) na educação.

Já no caso de *storytelling*, em [Hamilton and Weiss 2005], os autores defendem que o cérebro de um ser humano armazena informações em forma de histórias, pois elas ajudam a organizar e lembrar informações, conectando-as umas as outras. Por isso, se professores oferecem informações sem conexão para seus alunos, elas são facilmente esquecidas. Em [Green 2004] os autores afirmam que a contação de história é uma ferramenta única para apresentação de novas informações pois pode tornar a memória daquele fato inesquecível e divertido. Segundo [Green 2004], histórias podem ter várias funções em sala de aula como ”excitar o interesse dos alunos, ajudar o fluxo da aula, tornar o material apresentado memorável, superar a resistência ou ansiedade dos alunos e construir

uma relação entre o instrutor e os alunos assim como entre os próprios alunos".

O último conceito explorado é o ensino com base em projetos. Os autores em [Krajcik and Blumenfeld 2006] defendem que estudantes adquirem maior entendimento do material quando participam ativamente do entendimento e do uso das ideias. Nesse tipo de projeto estudantes se engajam em problemas de importância para eles e que são similares a problemas encontrados por cientistas, matemáticos, escritores etc. Estudantes são instigados a investigar as questões, propor hipóteses e explicações e discutir ideias. Ainda, segundo os autores, alunos do ensino com base em projetos possuem notas maiores que alunos do ensino tradicional.

No estudo apresentado neste artigo foram utilizados os conceitos apresentados anteriormente para a criação de um *framework* de ensino de programação (FEP) para crianças de idades variadas. É explicado o funcionamento básico deste *framework* de modo a apresentar os elementos que são importantes para a implementação do mesmo. Em seguida, é proposta uma arquitetura que descreve todas as relações entre os elementos do *framework*, provendo assim um método de aprendizado inclusivo e adequado a realidade atual.

2. FEP - *Framework* de Ensino de Programação

O *framework* proposto busca introduzir, desde a infância, novos paradigmas da era digital, despertando a atenção e motivação de seus usuários. Para isso, o FEP usa princípios do ensino baseado em projetos em que as crianças são componentes ativos de uma história ou atividade com o intuito de investigar e responder questões de níveis variados de dificuldade. Como o público-alvo são crianças de idades variadas, toda a interação com o FEP é feita por meio de computação tangível, ou seja, usa-se objetos que podem ser tocados e organizados como qualquer outro objeto já familiar à criança.

Os desafios propostos na história (que também pode ser vista como uma atividade) são então resolvidos por meio de algoritmos de computação com o objetivo do ensino de programação além de habilidades correlatas como raciocínio lógico, organização de pensamentos e ações, aprendizado de matemática e física, estimulo da criatividade e desenvolvimento de habilidades para solucionar situações adversas. Esses algoritmos são então executados em sistemas embarcados como *drones*, robôs, carros de controle remoto etc. Quando não existe o acesso a um sistema embarcado, é possível a executação do algoritmo dentro de um ambiente de simulação, uma vez que existem diversos ambientes de simulação para esse tipo de sistemas encontrados e disponibilizados de forma aberta e gratuita.

Por fim, por meio de dispositivos de Internet das Coisas como *smartphones* e *tablets* todos os alunos são interligados dentro da história, criando um grande ambiente de colaboração. Todas as atividades sendo feitas podem ser monitoradas pelo professor que consegue então acompanhar o progresso e delegar atividades personalizadas para cada aluno ou grupo de alunos.

3. Elementos do *framework*

Nessa seção são descritos os elementos necessários para a implementação do FEP. Embora para seu funcionamento completo seja necessária a implementação de todos os elementos em conjunto, uma análise individual de cada conceito é importante para a total

compreensão do *framework*. Os sete elementos essenciais são: resolução de problemas, *storytelling*, programação, visão *top-down* de ensino, colaboração/interação entre usuários, interação com o ambiente e baixo custo.

3.1. Resolução de problemas

Em um modelo de aprendizado tradicional sequencial baseado em memorização, grande parte do conteúdo ensinado é passado sem contextualização de modo que o aluno aprende momentaneamente mas, em geral, não sabe para que serve ou a importância daquele conhecimento. Isso pode gerar desinteresse, falta de engajamento e não fixação do conteúdo em sala de aula. No ensino com base em resolução de problemas, primeiro o contexto é dado na forma de questões a serem respondidas. Ao passo que os alunos precisam resolver os problemas dados, os conteúdos necessários são ensinados e imediatamente aplicados. Isso significa e implica em uma participação ativa dos alunos na construção de seu próprio conhecimento além de utilizarem na prática métodos científicos de investigação e resolução de problemas. Neste contexto, o ensino de programação por meio desta técnica de ensino é natural pois a computação é justamente um modo abstrato de modelar e resolver problemas.

Esses problemas podem ser de dois tipos:

- **Repetição**: um conjunto de passos é fornecido e a criança deve reproduzi usando a linguagem de programação. Esse exercício busca o ensino da sintaxe da linguagem de programação e a familiarização com esse tipo de abstração.
- **Elaboração**: um problema é dado e a criança deve usar seus conhecimentos e informações dadas para chegar em uma solução. Nesse tipo de exercício podem ser abordados assuntos diversos como lógica, matemática e física.

3.2. *Storytelling*

Segundo [Hamilton and Weiss 2005], o cérebro humano armazena informações em forma de histórias, pois elas ajudam a organizar e lembrar informações, conectando-as umas as outras. No ensino tradicional, essas informações são passadas sem conexão sendo então facilmente esquecidas. Neste *framework*, todos os problemas e conteúdos apresentados devem fazer parte de uma história em que as crianças são participantes ativos. Portanto, como citado na seção anterior, a criança aprende de modo ativo junto a um contexto que mostra o porquê aprender o conteúdo que está sendo ensinado.

Essas histórias devem ser dinâmicas e abordar assuntos com aplicações reais na sociedade. Isso é importante para mostrar para as crianças desde cedo que com conhecimento e dedicação é possível resolver os mais variados problemas existentes na sociedade.

3.3. Programação

O objetivo de programação é encontrar uma sequência de instruções que permitirá automatizar uma tarefa específica ou resolver um dado problema. Esse conceito pode ser aplicado para os mais variados domínios do conhecimento e constitui uma ótima forma de modelagem e resolução de problemas. Isso é tanto verdade que hoje é quase impossível encontrar uma área do conhecimento ou da indústria que não utiliza programação de algum modo. Portanto, o ensino de programação passa inicialmente pela transferência de conhecimento de modo que problemas e tarefas podem ser solucionadas eficientemente e chega a inclusão digital dos alunos que vivem em uma sociedade atual muito dependente de seus sistema de computação.

3.4. Visão *top-down* de ensino

A visão *top-down* é uma abordagem de pensamento e ensino em que o aprendizado inicia-se por uma visão geral, de modo que cada nível vai sendo detalhado até chegar em níveis mais básicos do elemento abordado. Essa visão foi escolhida para esse *framework*, pois o funcionamento de um sistema programado é de fácil entendimento quando abstraído. Porém, é de difícil compreensão quando pensado todos os detalhes que o envolvem tanto no âmbito de software quanto de hardware. Portanto, optou-se por uma abordagem prática inicialmente, usando uma linguagem de programação muito abstrata que possibilite a execução do código de forma imediata, sem uma curva de aprendizado grande. A cada nível que a criança vai aprendendo, os conceitos de programação são aprofundados até chegar, nos últimos módulos, no estudo de assuntos mais específicos como organização de computadores etc. Vale lembrar que mesmo em assuntos específicos as atividades devem ainda possuir uma abordagem prática que pode ser atingida por meio da utilização de infográficos interativos, por exemplo. Por fim, a visão *top-down* é importante para o FEP pois o que se busca é apresentar para as crianças o quanto se pode fazer com programação demonstrando sistemas funcionais e como elas são capazes de desenvolver esses sistemas ao passo que vão aprendendo os conceitos de programação.

3.5. Colaboração/interação entre usuários

Com o uso do FEP, as crianças terão acesso em conjunto a mesma história e cada aluno ou grupo de alunos fica responsável por um problema a ser resolvido. Cada problema tem algum tipo de relação com o outro, ou seja, os alunos devem trabalhar de modo colaborativo com o objetivo de ajudarem os personagens da história. Assim, todas as ações na história devem ser visualizadas em tempo real. Além da interação entre os alunos, o professor deve ser informado em tempo real de todas as ações dos seus alunos no sistema. Quando o problema envolve o controle de um sistema embarcado, os comandos são primeiro enviados ao professor que libera ou não sua execução no sistema embarcado. Isso é importante pois evita possíveis acidentes além de tornar viável a utilização de um número reduzido de sistemas embarcados em sala de aula, realidade de grande parte das escolas.

3.6. Interação com o ambiente

O FEP tem como ideia a aplicação imediata dos conhecimentos que são adquiridos, explorando o ambiente de modo que a computação não fique limitada a dispositivos eletrônicos considerados ”computadores”. A computação extrapola esses tipos de dispositivos que são apenas fins para a execução. A base é a lógica envolvida para transformar a solução de um problema em uma sequência de passos, conhecida como algoritmo. Além disso, para o uso de computadores é necessário o aprendizado de periféricos como *mouse* e teclado. Com isso, o FEP propõe a utilização de computação tangível, ou seja, a utilização de blocos físicos, como peças de quebra-cabeça, assim como as ilutradas na seção resultados, Figura 1. Esses blocos podem ser organizados de modo a formar sequências lógicas de comandos para resolver um problema, como ilustrado na seção resultados, Figura 2.

Um algoritmo gera a solução em forma de comandos (ilustrados na Figura 2), os quais devem ser executados em algum dispositivo. Levando em consideração a busca por interação com o ambiente, a utilização de robôs se mostra algo natural, além de conceitos de programação serem inerentes ao funcionamento de um robô.

3.7. Baixo custo

O FEP é ainda destinado a todas as classes sociais e busca oferecer inclusão digital para a maior quantidade possível de escolas. Isso só pode ser feito caso o acesso seja fácil, rápido e as soluções envolvidas criativas. Hoje no Brasil ainda é comum a falta de infraestrutura e treinamento de profissionais para a manipulação de sistemas de computação. Além disso, a criação de infraestruturas complexas tendem a gerar constantes problemas de manutenção que atrapalham o andamento das aulas e que tiram o foco das crianças. Por esses motivos, o FEP propõe a utilização de arquiteturas cliente-servidor locais e móveis de comunicação em que protocolos de tolerância a falhas são de grande importância. Esses dispositivos móveis podem ser simples, precisando de poucos pré-requisitos para o total funcionamento dentro da rede.

4. Resultados

Nas seções anteriores foi discutida a visão geral do FEP e seus componentes teóricos. Nessa seção são apresentados os resultados desse estudo divididos em: tecnologias e arquitetura. Vale lembrar que esse framework faz parte de uma pesquisa em andamento de modo que o mesmo ainda não foi completamente implementado e testado.

4.1. Tecnologias

Cada conceito essencial do *framework* pode ser implementado de formas diferentes e usando tecnologias diferentes. Buscando englobar todas as características propostas, as seguintes tecnologias foram escolhidas: *Wi-fi*, *bluetooth*, visão computacional, código de barras e dispositivos da Internet das Coisas.

4.1.1. Wi-fi

Com a tecnologia *Wi-fi* é possível manter todos os dispositivos presentes na arquitetura do *framework* em constante comunicação além de oferecer flexibilidade e escalonamento da rede. Sistemas sem fio são de fácil manuseio e evitam acidentes que poderiam ocorrer com dispositivos com fios. Ainda, é possível a implementação de redes tanto locais quanto remotas de modo rápido e sem levantar muitos problemas relacionados a infraestrutura [Lee et al. 2007], [Ferro and Potorti 2005].

4.1.2. Bluetooth

Esse protocolo é muito usado para a comunicação de sistemas embarcados como sensores e atuadores em geral, e pode ser usado em conjunto com uma rede *Wi-fi* já estabelecida. Além disso, um dispositivo usando tecnologia *bluetooth* pode se conectar a mais de uma rede ao mesmo tempo, enquanto que usando *Wi-fi* só é possível se conectar a uma por vez [Miller and Bisdikian 2001],[Lee et al. 2007], [Ferro and Potorti 2005].

4.2. Código de Barras

Com o uso da computação tangível, é necessário algum tipo de tecnologia para identificação dos objetos manipulados. Levando em consideração custo e simplicidade, optou-se por usar códigos de barras [Ebling and Cáceres 2010]. Esses códigos podem ser

facilmente impressos e distribuídos para os alunos em forma de blocos do tipo quebra-cabeça, conforme ilustrado na Figura 1. Nessa figura podem ser observados não só os códigos de barra mas também os comandos que eles representam. Essa parte do framework já foi implementada e testada. O programa desenvolvido gera automaticamente as peças e é destinado ao público final como professores e desenvolvedores.

Figure 1. Blocos da linguagem tangível proposta. Esses blocos foram desenvolvidos pelos pesquisadores deste artigo.

4.3. Visão Computacional

Para identificar os objetos pelos códigos de barras, foi utilizada a visão computacional que permite, por meio de *streamings* de imagens, encontrar e classificar cada código de barras e consequentemente o objeto correspondente [Nielsen 2005]. Essa parte do framework também já foi implementada e testada como ilustrado na Figura 2. Esse algoritmo foi implementado de modo que mais de um código de barras possa ser identificado ao mesmo tempo, assim não sendo necessário a leitura sequencial dos códigos de barras. Contudo, está em desenvolvimento o algoritmo responsável por identificar a relação de ordem e dependência das peças da liguagem tangível extraídas da imagem.

4.4. Dispositivos

Quanto aos dispositivos usados para a implementação do FEP tem-se: *smartphones*, *tablets*, sensores e atuadores (sistemas embarcados/robôs).

4.4.1. *Smartphones/tablets*

Como, nos dias de hoje, os *smartphones* e *tablets* fazem parte da vida da maioria das pessoas durante o tempo todo, é natural que esses dispositivos possam ser usados de modo central na implementação do FEP. Com esses dispositivos é possível tirar as fotos das peças da linguagem tangível, processar essas informações por meio de visão computacional, enviar os comandos para os sistemas embarcados diretamente ou para outros dispositivos, além de ser possível visualizar as história e os problemas, tudo em um único

dispositivo. Como citado nas seções anteriores, os algoritmos de visualização computacional para reconhecimento da linguagem tangível já foram implementados e são executados em smartphones com sistema operacional *Android*, como é possível observar na Figura 2.

4.4.2. Sensores e atuadores

Os sensores podem ser usados como uma forma de interação entre os problemas apresentados na história e o ambiente em volta da criança. Com isso é possível receber informações dos sensores que serão manipuladas pelos algoritmos criados pelos alunos e então executados em atuadores. Os atuadores são, em geral, sistemas embarcados que realizam tarefas específicas. Como atuadores tem-se *drones*, robôs, braços mecânicos, carrinhos e barcos de controle remoto entre outros. Tanto sensores quanto atuadores se comunicam com a rede do FEP por meio de *Bluetooth*, mas também podendo ser por meio de *Wi-fi*.

4.5. Arquitetura

4.5.1. Comunicação entre os *smartphones/tablets*

Em relação à comunicação entre os dispositivos do FEP, foi definida a criação de uma rede local com arquitetura cliente-servidor. O dispositivo do professor funciona como um servidor e roteador que distribui e recebe mensagens dos dispositivos dos alunos. O conteúdo das aulas são então obtidos da Internet pelo dispositivo do professor antes do começo da aula (isso pode ser inclusive obtido em algum outro lugar fora da sala de aula com acesso à Internet). Quando os alunos se conectam ao dispositivo do professor, esse conteúdo é então distribuido localmente para todos os alunos. Vale lembrar que o celular do professor funcionará como um roteador, dispensando então a configuração de uma rede local propriamente dita. O objetivo dessa configuração é assegurar a conectividade entre os dispositivos mesmo quando não houver Internet no local, realidade de boa parte das escolas principalmente públicas de ensino fundamental e médio no Brasil. Além disso, transmissão local de dados tende a ser muito mais rápida que transmissão remota de arquivos pela Internet.

4.5.2. Comunicação entre *smartphones/tablets* e sistemas embarcados (sensores e atuadores)

Quando um aluno terminar de programar uma atividade, ele terá a opção de executar aquele código em um sistema embarcado. Como já explicado nas seções anteriores, por questões de segurança, organização e aproveitamento de uma quantidade limitada de sistemas embarcados, todos os códigos a serem executados são enviados para o dispositivo do professor que então autoriza a execução ou não. Portanto, a conexão direta com os sistemas embarcados só ocorre com o dispositivo do professor. Isso é importante pois procedimentos de conexão, pareamento e configuração dos sistemas embarcados ficam limitadas ao dispositivo do professor.

4.5.3. Módulos, simulações e desenvolvedores

Para que o sistema como um todo se comunique com sistemas embarcados (sensores e atuadores) faz-se necessária a criação de um protocolo de comunicação para cada caso específico. Esses protocolos são implementados em formato de módulos do sistema que podem ser baixados pelo dispositivo do professor. Em outras palavras, o professor por meio de seu dispositivo tem acesso a uma biblioteca de módulos, cada um para um sistema embarcado específico. Por outro lado, esses módulos podem ser criados por uma comunidade de desenvolvedores onde é possível inserir seus projetos na plataforma *online*. Além dos módulos de comunicação, os desenvolvedores podem criar módulos de simulação que executam os comandos enviados pelo aluno dentro de uma plataforma semelhante a um jogo digital que representa o sistema embarcado.

4.5.4. Protocolos de leitura

Todos os programas são construídos por meio de blocos do tipo quebra-cabeça ilustrado na Figura 1 que são identificados posteriormente por *streaming* de imagens e transformados em código. Além disso, quando o professor cria no sistema uma sala, para cada aluno é gerado um bloco que possui um código de barras único. Esse código será identificado pelo dispositivo dos alunos e então todas as configurações necessárias serão feitas automaticamente. Em outras palavras, as crianças não terão que configurar ou digitar nada para já começarem a utilizar a plataforma. Um código completo gerado a partir dos comandos efetuados por meio dos blocos ilutrados na Figura 1 é ilustrado na Figura 2.

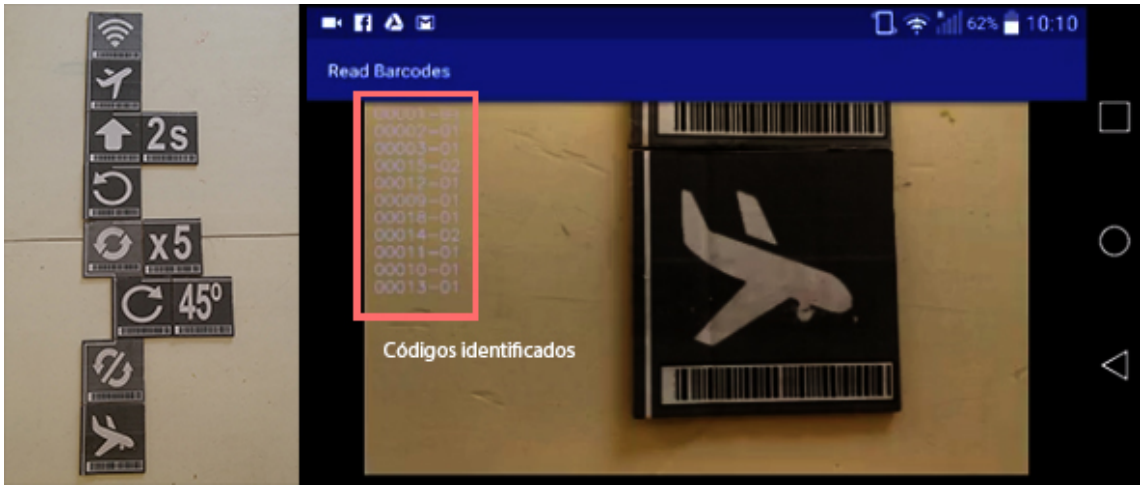

Figure 2. À esquerda, algoritmo criado por blocos de computação tangível. À direita, identificação desses blocos pelo sistema do aluno após percorrer todo o algoritmo.

4.5.5. Sistemas

O FEP quando implementado na sua totalidade deve possui três sistemas diferentes como mostrado na Figura 3. O primeiro é o sistema dos alunos. Esse sistema é destinado a

visualização das histórias e tarefas e identificação dos blocos de programação por meio da câmera do dispositivo. Além disso, caso necessário, pode simular a execução de um sistema embarcado. O segundo sistema é o do professor. Esse sistema é destinado ao *download* das histórias, tarefas, módulos e simulações, transmissão dos dados e acompanhamento das atividades realizadas pelos alunos além do recebimento e execução dos códigos em sistemas embarcados. O terceiro sistema é uma plataforma *online* que centraliza a comunidade de desenvolvedores, implementada por meio de nuvens ou outra tecnologia que viabilize o paradigma Internet das Coisas. Esses desenvolvedores por meio deste sistema podem publicar seus módulos, simulações, histórias e atividades que serão posteriomente baixadas pelos professores em seus dispositivos.

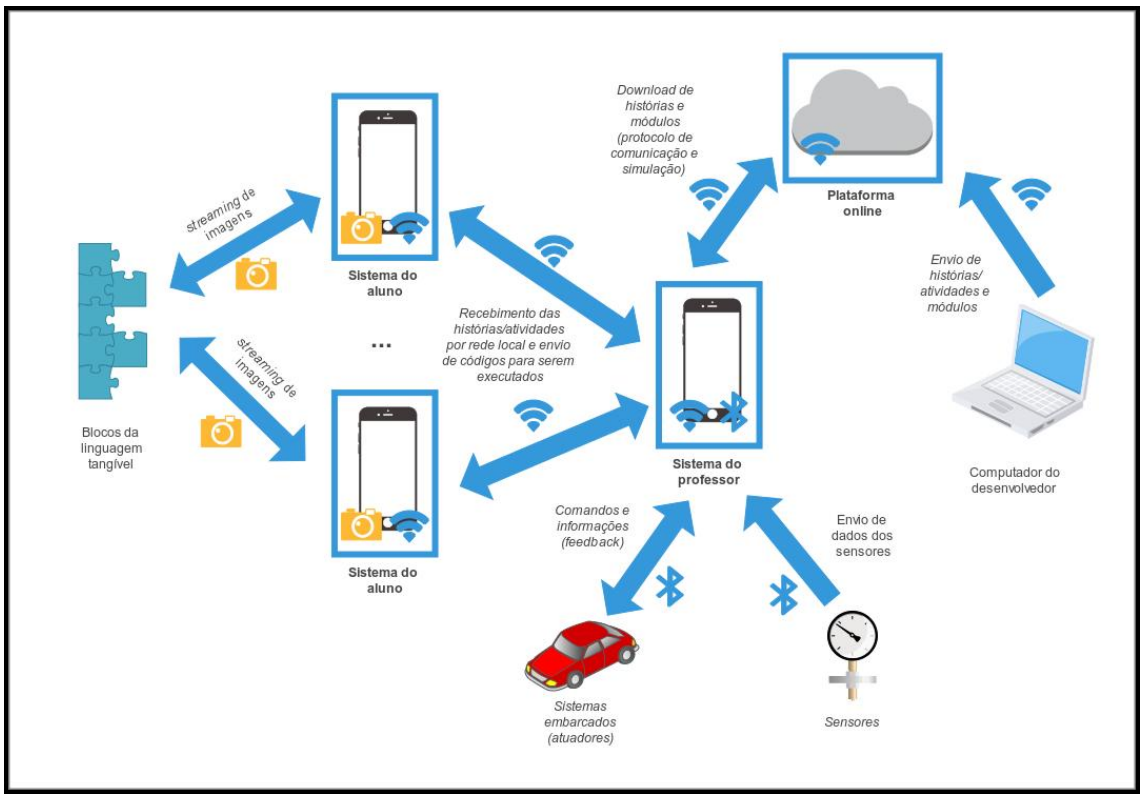

Figure 3. Esquemático demonstrando a relação entre cada sistema do FEP

5. Conclusões

Nesse estudo foi descrito um framework de ensino de programação (FEP) de baixo custo para crianças de variadas idades que está em atual implementação. Foram discutidos seus elementos teóricos como resolução de problemas, *storytelling*, programação, visão *top-down* de ensino, colaboração/interação entre usuários, interação com o ambiente e baixo custo. Além disso, foram discutidas as técnicas para aplicação dos elementos teóricos como computação tangível, sistemas embarcados e internet das coisas.

Também foram apresentados os resultados das análises de como o FEP pode ser implementado e quais partes do framework já foram implementadas e testadas. Foram descritas suas tecnologias, sistemas e arquiteturas que englobam toda a plataforma de ensino.

Como trabalho futuro pretende-se implementar a plataforma como um todo e analisar por meio de métricas de ensino o impacto que esse tipo de *framework* pode ter no ensino de programação para crianças.

6. Agradecimentos

Os autores gostariam de agradecer à Capes pelo apoio financeiro concedido pelo Programa Nacional de Cooperação Acadêmica (Procad) e à Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) pelo apoio financeiro concedido através do processo 2017/05925-3.

References

Bers, M. U., Flannery, L., Kazakoff, E., and Crouser, R. J. (2010). A curriculum unit on programming and robotics.

Bers, M. U. and Horn, M. S. (2010). Tangible programming in early childhood. *High-tech Tots: Childhood in a Digital World*, 49:49–70.

Ebling, M. and Cáceres, R. (2010). Bar codes everywhere you look. *IEEE Pervasive Computing*, 9(2):4–5.

Fagin, B. S., Merkle, L. D., and Eggers, T. W. (2001). Teaching computer science with robotics using ada/mindstorms 2.0. In *ACM SIGAda Ada Letters*, volume 21, pages 73–78. ACM.

Ferro, E. and Potorti, F. (2005). Bluetooth and wi-fi wireless protocols: a survey and a comparison. *IEEE Wireless Communications*, 12(1):12–26.

Fidalgo Neto, A., Tornaghi, A., Meirelles, R., Berçot, F., Xavier, L., Castro, M., and Alves, L. (2009). The use of computers in brazilian primary and secondary schools. *Computers & Education*, 53(3):677–685.

Green, M. C. (2004). Storytelling in teaching. *APS Observer*, 17(4):37–39.

Gregio, B. M. A. (2004). A informática na educação: As representações sociais e o grande desafio do professor frente ao novo paradigma educacional. *Revista Digital da CVA*, 2(6).

Hamilton, M. and Weiss, M. (2005). *Children tell stories: Teaching and using storytelling in the classroom*. Richard C Owen Pub.

Horn, M. S., Solovey, E. T., Crouser, R. J., and Jacob, R. J. (2009). Comparing the use of tangible and graphical programming languages for informal science education. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 975–984. ACM.

Krajcik, J. S. and Blumenfeld, P. C. (2006). *Project-based learning*. na.

Lee, J.-S., Su, Y.-W., and Shen, C.-C. (2007). A comparative study of wireless protocols: Bluetooth, uwb, zigbee, and wi-fi. In *Industrial Electronics Society, 2007. IECON 2007. 33rd Annual Conference of the IEEE*, pages 46–51. Ieee.

Miller, B. A. and Bisdikian, C. (2001). *Bluetooth revealed: the insider's guide to an open specification for global wireless communication*. Prentice Hall PTR.

Nielsen, F. (2005). *Visual computing: Geometry, graphics, and vision*. Charles River Media Hingham.

Radoslava, K., Velin, K., and Dafina, K. (2016). Investigating some programming languages for children to 8 years.

Smith, A. C., Springhorn, H., Mulligan, S. B., Weber, I., and Norris, J. (2011). tactuslogic: Programming using physical objects. In *IST-Africa Conference Proceedings, 2011*, pages 1–9. IEEE.