

UNIVERSIDADE DO ESTADO DO RIO GRANDE DO NORTE – UERN  
FACULDADE DE CIÊNCIAS EXATAS E NATURAIS – FANAT  
DEPARTAMENTO DE INFORMÁTICA – DI  
CURSO DE CIÊNCIA DA COMPUTAÇÃO

JOÃO MARINHO NETO

**SURVEY PARA CARACTERIZAR E CLASSIFICAR LINGUAGENS E AMBIENTES  
DE ENSINO DE PROGRAMAÇÃO PARA CRIANÇAS**

MOSSORÓ - RN

2016

JOÃO MARINHO NETO

**SURVEY PARA CARACTERIZAR E CLASSIFICAR LINGUAGENS E AMBIENTES  
DE ENSINO DE PROGRAMAÇÃO PARA CRIANÇAS**

Monografia apresentada à Universidade do Estado do Rio Grande do Norte como um dos pré-requisitos para obtenção do grau de bacharel em Ciência da Computação, sob orientação do Prof. M.Sc Antônio Oliveira Filho.

MOSSORÓ - RN

2016

**Catálogo da Publicação na Fonte.  
Universidade do Estado do Rio Grande do Norte.**

Marinho Neto, João

Survey Para Caracterizar E Classificar Linguagens E Ambientes De Ensino De Programação Para Crianças. / João Marinho Neto – Mossoró, RN, 2016.

60 f.

Orientador(a): Prof. Msc. Antônio Oliveira Filho.

Monografia (Bacharelado) Universidade do Estado do Rio Grande do Norte. Curso de Ciências da Computação

1. Revisão Sistemática. 2. Ensino de programação. 3. Ambientes Linguagens de programação. I. Antônio Oliveira Filho. II. Universidade do Estado do Rio Grande do Norte. III. Título.

UERN/ BC

CDD 004

Bibliotecário: Sebastião Lopes Galvão Neto – CRB - 15/486

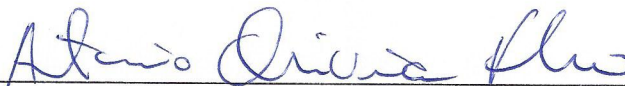
JOÃO MARINHO NETO

**SURVEY PARA CARACTERIZAR E CLASSIFICAR LINGUAGENS E AMBIENTES  
DE ENSINO DE PROGRAMAÇÃO PARA CRIANÇAS**

Monografia apresentada como pré-requisito para obtenção do título de Bacharel em Ciência da Computação da Universidade do Estado do Rio Grande do Norte – UERN, submetida à aprovação da banca examinadora composta pelos seguintes membros:

Aprovado em: 05 / 05 / 2016

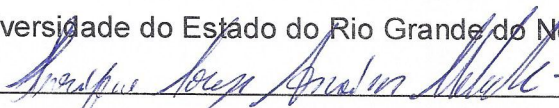
Banca Examinadora



---

Prof. M.Sc Antônio Oliveira Filho (Orientador)

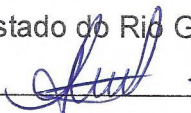
Universidade do Estado do Rio Grande do Norte – UERN



---

Prof. D.Sc Henrique Jorge Amorim Holanda

Universidade do Estado do Rio Grande do Norte – UERN



---

Prof. M.Sc Jessica Neiva de Figueiredo Leite

Universidade do Estado do Rio Grande do Norte – UERN

*A meus pais que sempre me apoiaram  
e me deram suporte nessa jornada*

## **AGRADECIMENTOS**

Agradeço primeiramente a Deus, pois para ele tudo é possível.

Agradeço todo o apoio dos meus pais Brasiliano e Girlene, que sempre me apoiaram não importando as circunstâncias e me mostram que as dificuldades podem ser superadas, não faltando amor e compressão em todas as minhas decisões.

Aos ComPuteiros (Wedson, Thomaz, Chrystian, Arthur, Erick e Ramon), que considero mais que amigos, são meus irmãos. Obrigado por tornar a graduação muito mais divertida e os problemas menores. Obrigado pelas festas e os momentos inesquecíveis e por nunca me deixarem na mão quando precisei.

O que não provoca minha morte faz  
com que eu fique mais forte.

Friedrich Nietzsche.

## RESUMO

A programação é vista como uma atividade técnica e restrita à um pequeno segmento da população. Isso tem ocasionado problemas como déficit de profissionais qualificados na área de TI e um número alto de desistência em cursos voltados a área de computação. É necessário, portanto buscar um modo de tornar o ensino de programação mais abrangentes e acessível para tipos diferentes de usuários. Iniciativas apoiadas por grandes nomes da tecnologia como Bill Gates e Mark Zuckerberg têm o intuito de tornar o aprendizado da lógica programacional mais fácil e intuitiva ainda na infância. A Intenção deste trabalho é fazer uma análise de ambientes e linguagens que ajudem o aprendizado de programação por crianças e pessoas que nunca tiveram contato com uma linguagem de programação. No final deste trabalho é proposto um modo de classificação dessas ferramentas de ensino de programação, por meio de informações coletadas através de uma revisão sistemática.

**Palavras-chave:** Revisão Sistemática, Ensino de programação, Ambientes, Linguagens de programação.



## ABSTRACT

The programming is been seen as a technical activity and restricted to a small segment of population. This has caused problems such as shortage of qualified professionals in the IT field, and a high number of dropout in courses focused computing area. It is therefore necessary to seek a way to make the teach programming more inclusive and accessible to different types of users. Initiatives supported by big names in technology like Bill Gates and Mark Zuckerberg are intended to make learning easier and intuitive programacional logic in childhood.. The objective of this study is to analyze environments and languages that help the learning programming for children and people who have never had contact with a programming language. The end of this paper proposed a method of classification of these programming teaching tools, using information collected through a systematic review.

**Keywords:** Systematic review, Programming education, Environments, Languages programming.

## LISTA DE SIGLAS

BRASSCOM	Associação Brasileira de Empresas de Tecnologia da Informação e Comunicação
SBC	Sociedade Brasileira de Computação
TI	Tecnologia da Informação

## LISTA DE FIGURAS

Figura 3.1: Hierarquia de Evidências .....	18
Figura 4.1: Mecanismos de Ensino de Programação.....	36
Figura 4.2: Interface Inicial do Alice .....	38
Figura 4.3: Tela Principal do Scratch .....	39
Figura 4.4: Tela do AgentSheets.....	40
Figura 4.5: Tela do Kodu.....	41
Figura 4.6: Tela Inicial do Squeak Etoys .....	42
Figura 4.7: Tela do RoboMind.....	43
Figura 4.8: Tela Inicial do Greenfoot .....	44
Figura 4.9: Tela Inicial do Stagecast Creator .....	45
Figura 4.10: Tela do Lightbot.....	46
Figura 4.11: Tela do Blockly .....	47
Figura 4.12: Kodable Sendo Executado.....	48
Figura 4.13: Ambiente Code.org .....	49

## LISTA DE QUADROS

Quadro 3.1.1 .....	21
Quadro 3.1.2 .....	21
Quadro 3.1.3 .....	21
Quadro 3.2.1 .....	21
Quadro 3.2.2 .....	21
Quadro 3.2.3 .....	21

## LISTA DE TABELAS

Tabela 3.1: Seleção de Artigos .....	22
Tabela 4.1: Características Gerais das Ferramentas Avaliadas.....	50
Tabela 4.2: Paradigma de Programação.....	51
Tabela 4.3: Estruturas de Programação Suportadas .....	52
Tabela 4.4: Representação do Código.....	53
Tabela 4.5: Construção de Programas.....	53
Tabela 4.6: Método de Ensino.....	54
Tabela 4.7: Ferramentas por Idade e Nível .....	55

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO .....</b>	<b>13</b>
<b>2</b>	<b>REFERENCIAL TEÓRICO .....</b>	<b>14</b>
2.1	DIFICULDADES NO APRENDIZADO DE PROGRAMAÇÃO .....	14
2.2	LOGO: O PRIMEIRO SOFTWARE EDUCACIONAL.....	15
<b>3</b>	<b>REVISÃO SISTEMÁTICA .....</b>	<b>17</b>
3.1	METODOLOGIA .....	18
3.2	PROCESSO DA REVISÃO SISTEMÁTICA .....	19
<b>3.2.1</b>	<b>Questão de Pesquisa .....</b>	<b>19</b>
<b>3.2.2</b>	<b>Fontes de Pesquisa .....</b>	<b>20</b>
<b>3.2.3</b>	<b>String de Busca .....</b>	<b>20</b>
<b>3.2.4</b>	<b>Palavras-Chave.....</b>	<b>20</b>
<b>3.2.5</b>	<b>Restrições .....</b>	<b>21</b>
<b>3.2.6</b>	<b>Execução das Buscas .....</b>	<b>22</b>
3.3	RESULTADO DAS BUSCAS.....	26
<b>4</b>	<b>LINGUAGENS E AMBIENTES DE ENSINO DE PROGRAMAÇÃO PARA CRIANÇAS .....</b>	<b>34</b>
4.1	METODOLOGIA DOS SISTEMAS DE ENSINO DE PROGRAMAÇÃO .....	35
<b>4.1.1</b>	<b>Criação, Estruturação e Execução.....</b>	<b>36</b>
4.2	CARACTERÍSTICAS DAS LINGUAGENS E AMBIENTES DE PROGRAMAÇÃO .....	37
4.3	ANÁLISE DAS FERRAMENTAS.....	49
<b>4.3.1</b>	<b>Conceitos Avaliados das Ferramentas.....</b>	<b>51</b>
<b>5</b>	<b>CONSIDERAÇÕES FINAIS.....</b>	<b>56</b>
	<b>REFERÊNCIAS.....</b>	<b>57</b>

## 1 INTRODUÇÃO

Iniciativas e ferramentas tem surgido ao logo dos anos com o intuito de facilitar o ensino de programação a crianças. O processo de aprendizagem de uma linguagem de programação na infância tem sido muito defendido por grandes nomes da tecnologia como Mark Zuckerberg e Bill Gates e segundo eles, sendo sua importância tão fundamental e básica quanto ler e escrever.

Partindo desse pressuposto o objetivo deste trabalho é fazer um levantamento das ferramentas que apoiam o ensino de programação a crianças e iniciantes como forma de desenvolvimento do pensamento lógico. Será feito a análise dessas ferramentas buscando coisas em comum que possam ajudar no processo de aprendizado de uma linguagem de programação por parte das crianças. Esse estudo busca não só descrever características dessas ferramentas como também desenvolver um modo de classifica-las para encontrar a melhor maneira de usar essas ferramentas.

Esse trabalho foi proposto visto a relevância deste assunto e como ele vem sendo muito discutido atualmente, pelo fato de ser importante ter um pensamento lógico e saber uma linguagem de programação. Outro problema que podemos considerar é a evasão de cursos de computação associados a carência de uma base educacional na área de programação. Saber qual a ferramenta melhor se encaixa para crianças de idades diferentes pode ajudar a solucionar estes problemas e auxiliar o ensino de conceitos computacionais.

O documento está dividido em capítulos, sendo o capítulo 1 referente a parte introdutória do mesmo. No capítulo 2 é feito um levantamento teórico, para que o trabalho possa ser desenvolvido. O capítulo 3 descreve como foi feito a revisão sistemática, qual o processo utilizado para se obter material relevante para a caracterização e classificação das ferramentas de ensino de programação. No capítulo 4 é mostrado a metodologia dos sistemas de ensino de programação a iniciantes, logo após é feita uma descrição das ferramentas escolhidas no processo de leitura dos trabalhos selecionados na revisão sistemática. Depois disso é feita uma análise das ferramentas e proposto um método de classifica-las. No capítulo 5 são feitas as considerações finais deste trabalho.

## 2 REFERENCIAL TEÓRICO

### 2.1 DIFICULDADES NO APRENDIZADO DE PROGRAMAÇÃO

É notável a existência de uma grande dificuldade em compreender e aplicar conceitos abstratos de programação por parte dos alunos, que frequentam disciplinas introdutórias na área de computação (LAHTINEN et al. 2005).

Essa dificuldade em disciplinas de programação pode ser atribuída ao fato dessa atividade ser vista como restrita a um segmento da população e não ser vista antes na educação básica (RIBEIRO, 2012).

Segundo Kelleher e Pausch (2003) aprender a programar pode ser muito difícil para iniciantes de qualquer idade, e essa dificuldade tornar-se ainda maior quando se trata de crianças. São muitas as barreiras que colocam dificuldade ao ensino da programação a crianças como:

- **Estruturar a solução** – saber como solucionar os problemas de forma estruturada e bem definida;
- **Execução**: Aprender como os programas são executados;
- **Sintaxe**: Aprender os comandos das linguagens;
- **Desmotivação**: Programar não é vista como uma atividade divertida.

Para solucionar esse problema tem surgido ao longo das décadas ferramentais e atualmente diversas iniciativas que defendem o ensino de programação nas escolas regulares, sem restrições. Personalidades da tecnologia têm defendido publicamente o ensino de programação como forma de inclusão digital. Segundo eles, interpretar e escrever códigos é tão importante quanto ler e escreve. O aprendizado dessas linguagens ajuda na autonomia na hora de resolver problemas, incentiva o trabalho colaborativo e aumenta a capacidade de pensar de forma sistematizada e criativa. (GERALDES, 2014)

Segundo a Sociedade Brasileira de Computação (SBC) a computação é tão importante que “deve ser ensinada desde o ensino fundamental, assim como as ciências Física e Matemática”. Para Silva et al (2014; apud França et al. 2013) a computação “pode contribuir, de maneira interdisciplinar, na busca de soluções de



problemas diversos, através da disseminação do chamado pensamento computacional”.

Pensamento computacional é saber usar o computador como um instrumento de aumento do poder cognitivo e operacional humano, aumentando a nossa produtividade, inventividade e criatividade (FRANÇA, SILVA E AMARAL 2012). Para França, Silva e Amaral (2012; apud Nunes, 2011) é fundamental a introdução de conceitos de programação na educação básica como forma de aprimorar o raciocínio computacional das crianças, pelo seu caráter transversal a todas as ciências.

Outro problema que essas iniciativas de ensino de programação tentam minimizar é a falta de profissionais qualificados na área de TI, segundo Associação Brasileira de Empresas de Tecnologia da Informação e Comunicação (BRASSCOM) O mercado de TI brasileiro apresenta um vasto déficit de profissionais, de acordo com um levantamento realizado em 2014 cerca de 33 mil vagas foram preenchidas por profissionais formados nessa área das 78 mil vagas ofertadas (SILVA et al, 2014).

## 2.2 LOGO: O PRIMEIRO SOFTWARE EDUCACIONAL

Nos anos de 1967 e 1968, o matemático Seymour Papert e sua equipe desenvolveram uma linguagem de programação totalmente voltada para a educação, chamada Logo. Se olharmos para ela do lado educacional, é uma linguagem simples, pois possui características que tornam acessível o seu uso por qualquer de pessoa de qualquer idade. Computacionalmente, LOGO é considerada bastante sofisticada. A linguagem LOGO tem caráter imperativo e é baseada na lógica formal, ou seja, o usuário comanda a máquina indicando o que o ela deve fazer, sem duplicidade de sentido e de forma ordenada. Para que isso seja possível o usuário precisa analisar as ações necessárias para cada tarefa, já que existem várias soluções possíveis (PRADO 2000, apud GERALDES, 2014).

O ambiente em torno do LOGO prioriza o uso uma “pedagogia de projetos”, onde as diversas áreas do conhecimento podem ser integradas na resolução de diferentes problemas, numa atitude cooperativa do grupo, catalisada pelo professor. Assim, essa linguagem oferece ao docente a possibilidade de acompanhar, passo a passo, o raciocínio lógico da criança e analisar o que ela fez. Como catalisador, é preciso que espere o tempo de cada sujeito. Essa vivência desperta na criança a

responsabilidade sobre seu desenvolvimento, a segurança diante de situações desconhecidas, além de levá-la a refletir sobre seu próprio conhecimento. (MARTINS, 2012)

A linguagem LOGO foi criada a partir de um conceito também criado por Papert o “construcionismo”, segundo ele, uma abordagem pela qual o aprendiz constrói, por intermédio do computador, seu próprio conhecimento. Sendo uma reconstrução do construtivismo piagetiano, Papert concorda com o matemático Piaget, quando este afirma que a criança é um “ser pensante” e construtora de suas próprias estruturas cognitivas, mesmo sem ser ensinada. A atitude construcionista implica a meta de ensinar de forma a produzir o máximo de aprendizagem, com o mínimo de ensino. O construcionismo busca alcançar meios de aprendizagem forte que valoriza a construção mental do sujeito, libertando seu pensamento criativo. (MARTINS 2012, GERALDES 2014)

A palavra “LOGO” foi usada como referência a um termo grego que significa “pensamento, raciocínio e discurso”, ou, também, “razão, cálculo e linguagem”, com alusão à maneira livre e criativa pela qual a matemática é implementada para resolver problemas em forma de algoritmos. (MARTINS, 2012)

Segundo Jesus e Raabe (2014) LOGO não é apenas uma linguagem de programação, mas também uma filosofia. De acordo com a sua teoria do conhecimento e do desenvolvimento humano, o processo de aprendizado de uma criança não acontece apenas de modo formal, ela é um aprendiz inata, que assimila o conhecimento de modo espontâneo e intuitivo, que se dá através da exploração, e da investigação e pode ser caracterizado como uma real autoaprendizagem. Aquilo que uma criança aprendeu porque fez, reveste de um significado especial aquilo que foi aprendido ajudando a reter o conhecimento com mais facilidade. Na filosofia do LOGO está embutida a ideia que a aquisição de conhecimento não se dá em função do desenvolvimento, mas principalmente na maneira pela qual as pessoas se relacionam com o meio, ou seja, as condições que este oferece para exercitar o pensamento qualitativo.

Ao trabalhar com a linguagem LOGO, o erro é tratado como uma tentativa de acerto, ou seja, uma fase necessária à uma nova estrutura cognitiva. Essa linguagem pode ser usada por alunos de todas as idades, ou por qualquer pessoa interessada em “criar e construir seu conhecimento”. O ambiente tradicionalmente envolve uma tartaruga gráfica, um robô pronto para responder aos comandos do usuário. Uma vez

que a linguagem é interpretada e interativa, o resultado é mostrado, imediatamente, após digitar-se o comando incentivando o aprendizado. Se por exemplo algo está errado em seu raciocínio, isso é claramente percebido e demonstrado na tela, num processo de feedback, fazendo com que o aluno pense o que poderia estar errado e tente, com base nos erros vistos, encontrar soluções corretas para os problemas. (MARTINS, 2012)

Para Silva (2007) as principais características da linguagem LOGO que podemos destacar são:

- **Amigabilidade:** É uma linguagem de fácil aprendizado.
- **Modularidade:** É possível criar novos comandos para a linguagem usando a própria linguagem.
- **Interatividade:** Oferece uma resposta imediata e mensagens informativas sobre o comando aplicado.
- **Flexibilidade:** Pode ser usado com crianças de idades diferentes.
- **Capacidade:** possui ferramentas para criar programas com diversos graus de sofisticação.

### 3 REVISÃO SISTEMÁTICA

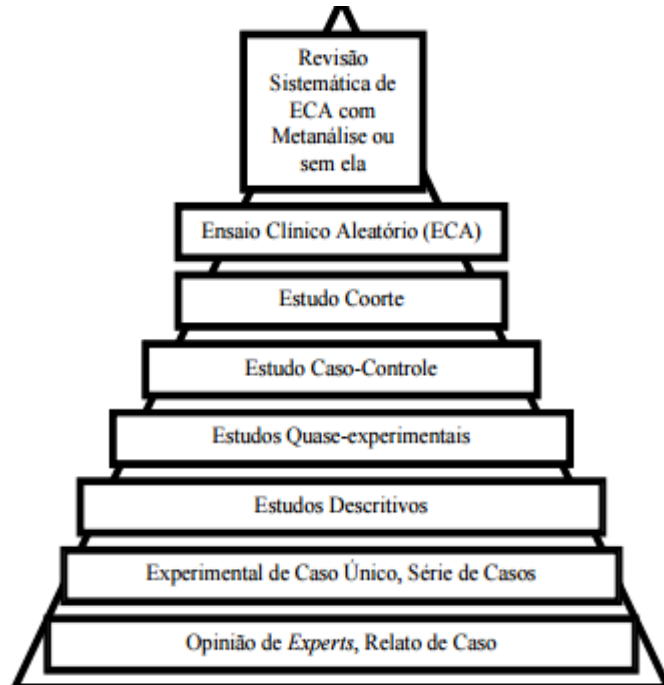
Segundo Sampaio e Mancini (2006) uma revisão sistemática, é uma forma de pesquisa que utiliza como fonte de dados a literatura sobre determinado tema. Como pode ser visto na figura 3.1, esse tipo de investigação disponibiliza um resumo das evidências relacionadas a uma estratégia de intervenção específica, mediante a aplicação de métodos sistematizados de busca, apreciação crítica e síntese da informação selecionada.

Uma revisão sistemática da literatura é um meio de identificar, avaliar e interpretar todas as pesquisas disponíveis relevantes para uma determinada questão de pesquisa ou área de assunto, ou fenômeno de interesse. Os estudos individuais que contribuem para uma revisão sistemática são chamados estudos primários; uma revisão sistemática é uma forma de um estudo secundário. (KITCHENHAM, 2004)

Antes de se iniciar uma revisão sistemática, precisamos levar em consideração três etapas: definir o objetivo da revisão, identificar a literatura e selecionar os estudos possíveis de serem incluídos. Essas etapas preliminares são importantes, uma vez

que ajudam a ter uma base norteadora da revisão com base na informação disponível sobre o tema de interesse (SAMPAIO & MANCINI, 2006).

**Figura 3.1:** Hierarquia de Evidências



**Fonte:** Sampaio e Mancini (2006)

### 3.1 METODOLOGIA

A Revisão Sistemática deste trabalho segue o modelo proposto por Kitchenham (2004), em que a revisão sistemática é dividida em três etapas: Planejamento, Condução e Análise dos resultados obtidos.

Na fase inicial de planejamento, são estabelecidos os objetivos da pesquisa e o desenvolvimento do protocolo de pesquisa. Nessa fase alguns componentes avaliados são: formulação de uma questão de pesquisa; fontes de buscas e seleção dos estudos (LEITE, 2011).

A etapa dois consiste na condução dos passos estabelecidos anteriormente. A string de busca será executada nas fontes de buscas e serão selecionados os artigos que atenderem as especificações.

Por último, será gerado um relatório com os resultados obtidos, que serão sintetizados e discutidos.

Primeiramente foram feitas pesquisas manuais referentes ao alvo da pesquisa para recolhimento de informações sobre essa área específica. Com essas pesquisas foram encontrados os trabalhos de Valeria e França (2016), no qual é realizado um

estudo comparativo de ferramentas de auxílio ao aprendizado de programação; e Medeiros, Silva e Aranha (2013) onde é feito uma revisão sistemática sobre o ensino de programação usando jogos digitais.

Para o estudo relacionado a revisão sistemática, foram usados os trabalhos de Kitchenham (2004) como base pelo fato da vasta experiência da autora nesse assunto e Sampaio e Mancini (2006) por realizarem um estudo sobre o modo de seguir numa revisão sistemática criteriosa e o mesmo serviu de modelo para a elaboração da revisão sistemática.

### 3.2 PROCESSO DA REVISÃO SISTEMÁTICA

O objetivo dessa revisão sistemática é fazer um levantamento de documentos que tenham abordado o tema proposto neste trabalho, assim como ampliar os estudos e a utilização desse tema.

Para a String de busca foi utilizado o modelo de Kitchenham (2004) onde é definido três componentes básicos para a estrutura: população, intervenção e resultados.

- População: conjunto afetado pela intervenção, área de aplicação.
- Intervenção: busca em específico na população estudada.
- Resultados: fatos a serem utilizados na comparação das intervenções.

#### 3.2.1 Questão de Pesquisa

Quais os ambientes e ferramentas relevantes para ensino de programação para crianças?

**População:**

Estudos de modelos de ensino de programação

**Intervenção:**

Ferramentas de ensino de programação a crianças

**Resultados:**

Trabalhos relacionados a esse assunto.

### 3.2.2 Fontes de Pesquisa

As bases eletrônicas foram baseadas de acordo com Dieste et al. (2009), que estabelece critérios como, cobertura das publicações, disponibilidade dos estudos primários em buscas em inglês; e em português para recuperar algum estudo importante no Brasil sobre esse assunto em questão. Sendo assim as máquinas de buscas digitais utilizadas foram:

- ACM Digital Library
- IEEE Xplore
- Springer
- Scopus
- Portal de Periódicos CAPES/MEC.

### 3.2.3 String de Busca

A String de busca é constituída pela junção das principais palavras chaves usando o conector “OR” para combinar sinônimos e outros termos, e o conector “AND” para combinar as palavras-chaves Zem-Lopes et al (2013; Apud KITCHENHAM, 2006).

### 3.2.4 Palavras-Chave

#### **População**

**Inglês:** Programming to kids.

**Português:** Programação para crianças.

#### **Intervenção**

**Inglês:** Classification, Description, Tools.

**Português:** Classificação, Caracterização, Ferramentas.

#### **Resultados:**

**Inglês:** Environments, Method.

**Português:** Ambientes, Métodos.

**String de busca em Inglês:****Quadro 3.1.1**

**String1:** (“Programming to Kids”) AND (“Programming to Kids Classification”) AND (“Environments” OR “Method”).

**Quadro 3.1.2**

**String2:** (“Programming to Kids”) AND (“Programming to Kids description”) AND (“Environments” OR “Method”).

**Quadro 3.1.3**

**String3:** (“Programming to Kids”) AND (“Programming to Kids Tools”) AND (“Environments” OR “Method”).

**String de busca em Português:****Quadro 3.2.1**

**String1:** (“Programação para Crianças”) AND (“Programação para crianças Classificação”) AND (“Ambientes” OR “Métodos”).

**Quadro 3.2.2**

**String2:** (“Programação para Crianças”) AND (“Programação para crianças Descrição”) AND (“Ambientes” OR “Métodos”).

**Quadro 3.2.3**

**String3:** (“Programação para Crianças”) AND (“Programação para Crianças Ferramentas”) AND (“Ambientes” OR “Métodos”).

**3.2.5 Restrições**

Os critérios de restrição são estabelecidos para garantir a seleção de estudos relevantes e relacionados a proposta da pesquisa.

Para Kitchenham (2004) os critérios de seleção são responsáveis para identificar os estudos primários que fornecem evidências sobre a questão da pesquisa. A fim de reduzir probabilidades de opiniões diferentes.

- **Critério 1 (C1):** Os documentos devem estar disponíveis em bibliotecas digitais, esse critério garante a disponibilidade do material.
- **Critério 2 (C2):** Os documentos devem estar escritos em inglês ou português.
- **Critério 3 (C3):** Os trabalhos selecionados devem abordar linguagens e ambientes de ensino de programação para crianças, facilitando a aprendizagem e os primeiros contatos com uma linguagem de programação.

### 3.2.6 Execução das Buscas

Nesta seção são apresentados os resultados das Strings de busca nas fontes selecionadas, embora inicialmente tenham sido selecionadas cinco fontes de pesquisa (ACM, IEEEExplore, Scopus, Spriger e o Portal de Periódicos da CAPES/MEC), devido às restrições impostas pelas bibliotecas de só disponibilizar resumos para universidades que não tem contrato com as mesmas, as buscas foram realizadas apenas no portal de periódicos da CAPES/MEC.

Como mostra na tabela 3.1, na execução das buscas foram encontrados 45 documentos, e através da leitura dos mesmo foi feito a seleção dos documentos de acordo com as restrições impostas no tópico 3.2.5.

**Tabela 3.1:** Seleção de Artigos

TÍTULO	C1	C2	C3	STATUS DO TRABALHO
Análise de ferramentas para o ensino de programação na Educação Básica	OK	OK	OK	SELECIONADO
Ensino de ciência da computação na educação básica: ferramentas, desafios e possibilidades	OK	OK	OK	SELECIONADO
Ferramentas de auxílio ao aprendizado de programação: um estudo comparativo	OK	OK	OK	SELECIONADO
Learning and teaching programming: A review and Discussion	OK	OK	OK	SELECIONADO



Jogo sério colaborativo para o ensino da programação a crianças	OK	OK	OK	SELECIONADO
Mindstorms: children, computers, and powerful ideas	OK	OK	X	EXCLUÍDO
Linguagem LOGO e a construção do conhecimento	OK	OK	X	EXCLUÍDO
Adequação de software educativo e formação continuada	OK	OK	X	EXCLUÍDO
Usando o SCRATCH para potencializar o pensamento criativo em crianças do ensino fundamental	OK	OK	OK	SELECIONADO
O ensino de lógica de programação e o desenvolvimento de jogos educacionais: um caso aplicado aos alunos do curso de licenciatura plena em matemática.	OK	OK	X	EXCLUÍDO
O uso de linguagem LOGO na educação infantil.	OK	OK	OK	SELECIONADO
Programar é bom para crianças? Uma visão crítica sobre o ensino de programação nas escolas.	OK	OK	OK	SELECIONADO
Programação de computadores como meio de desenvolvimento do raciocínio lógico em crianças e adolescentes.	OK	OK	OK	SELECIONADO
RoboEduc: especificação de um software educacional para ensino da robótica as crianças como uma ferramenta de inclusão digital.	OK	OK	OK	SELECIONADO
Ensino de programação utilizando jogos digitais: uma revisão sistemática da literatura.	OK	OK	OK	SELECIONADO
Avaliação empírica da utilização de um jogo para auxiliar a aprendizagem de programação.	OK	OK	OK	SELECIONADO

Ensinando princípios básicos de programação utilizando jogos educativos em um programa de inclusão digital.	OK	OK	X	EXCLUÍDO
Ensino de algoritmo a nível médio utilizando música e robótica: uma abordagem lúdica.	OK	OK	X	EXCLUÍDO
Kodu Game Labs: estimulando o raciocínio lógico através de jogos	OK	OK	X	EXCLUÍDO
Aprendendo a ensinar programação combinando jogos e python.	OK	OK	X	EXCLUÍDO
ProGame: um jogo para o ensino de algoritmos e programação.	OK	OK	X	EXCLUÍDO
Limitações da utilização do Alice no ensino de programação para alunos de graduação.	OK	OK	X	EXCLUÍDO
Proposta metodológica para a inserção ao ensino de lógica de programação com Logo e Lego Mindstorms	OK	OK	X	EXCLUÍDO
Atraindo alunos do ensino médio para a programação: uma experiência pratica de introdução a programação utilizando jogos e Python.	OK	OK	X	EXCLUÍDO
Programação no ensino médio: uma abordagem de ensino orientado ao design com Scratch.	OK	OK	X	EXCLUÍDO
Proposta de sistema lúdico para ensino de programação a alunos do ensino médio.	OK	OK	X	EXCLUÍDO
A robótica como ferramenta de apoio ao ensino de disciplinas de programação em cursos de computação e engenharia	OK	OK	X	EXCLUÍDO
Modelação de sistemas experimentais em ToonTalk para ambientes educativos: ciclo da água	OK	OK	OK	SELECIONADO

Limitações da Utilização do Alice no Ensino de Programação para Alunos de Graduação	OK	OK	OK	SELECIONADO
RoboEduc: um software para programação em níveis.	OK	OK	X	EXCLUÍDO
Lowering the Barriers to Programming: a survey of programming environments and languages for novice programmers	OK	OK	OK	SELECIONADO
Produzindo animações através da programação por demonstração.	OK	OK	OK	SELECIONADO
Uma visão do cenário Nacional do Ensino de Algoritmos e Programação: uma proposta baseada no Paradigma de Programação Visual	OK	OK	X	EXCLUÍDO
Ensino de Ciência da Computação na Educação Básica: Experiências, Desafios e Possibilidades	OK	OK	X	EXCLUÍDO
O uso do Lego Mindstorms no apoio ao Ensino de Programação de Computadores	OK	OK	X	EXCLUÍDO
Uma abordagem lúdica para a aprendizagem de programação de computadores	OK	OK	X	EXCLUÍDO
Ensino de Lógica de Programação a Crianças de 5º Ano da Escola de URI, Utilizando a Metodologia do Code Club Brasil e a Linguagem Scratch, para Incremento no Desenvolvimento e Rendimento Escolar	OK	OK	X	EXCLUÍDO
Um jogo para o ensino de programação em Python baseado na taxonomia de Bloom	OK	OK	X	EXCLUÍDO
Uma Proposta de Ensino-aprendizagem de Programação Utilizando Robótica Educativa e Storytelling	OK	OK	X	EXCLUÍDO

Using ToonTalk™ in Kindergartens	OK	OK	X	EXCLUÍDO
Takkou: Uma Ferramenta Proposta ao Ensino de Algoritmos	OK	OK	X	EXCLUÍDO
Relato de experiência da utilização da ferramenta Lightbot enquanto construtor do raciocínio para lógica de programação	OK	OK	OK	SELECIONADO
LogicBlocks: Uma ferramenta para o ensino de lógica de programação	OK	OK	OK	SELECIONADO
Análise da ferramenta de programação visual Blockly como recurso educacional no ensino de programação	OK	OK	OK	SELECIONADO
MODEBOTS: Environment for Programming Robots for Children Between the Ages of 4 and 6	OK	OK	OK	SELECIONADO
Um estudo exploratório dos games para introdução ao pensamento computacional.	OK	OK	OK	SELECIONADO

Como apresentado na tabela 3.1, foram encontrados 45 trabalhos de maior relevância por meio das Strings de busca:

- 21 trabalhos selecionados;
- 24 trabalhos excluídos.

A maioria dos trabalhos que foram excluídos fugiam da proposta dessa pesquisa ou não tinham informação relevantes para que se pudesse usar na classificação das ferramentas de ensino de programação para crianças.

### 3.3 RESULTADO DAS BUSCAS

Nessa fase os documentos selecionados pela revisão sistemática em questão serão apresentados com mais detalhes, mostrando a importância do mesmo para a pesquisa que está sendo desenvolvida.

Os trabalhos serão fundamentais para o levantamento das linguagens e ambientes de programação que melhor se adequam ao ensino a crianças e o quanto podem se desenvolver para que possam prosseguir a um nível mais avançado de linguagens de programação.

### **3.3.1 Análise de ferramentas para o ensino de programação na Educação Básica**

**Autor(es):** Eraylson Galdino da Silva, Aline F. Barbosa, Sebastião R. S. Neto, Renato H. O.Lopes, Ariane N. Rodrigues

**Fonte da Pesquisa:** Portal de Periódicos CAPES/MEC

**Ano de Publicação da Obra:** 2014

**Resumo/Abstract da Obra:** A inserção de conceitos de lógica de programação na Educação Básica tem sido emergente, visto que os processos de ensino atuais visam as mudanças tecnológicas e que aplicar a computação na Educação Básica é uma forma de motivar os alunos para o mercado de TI. Neste sentido, o presente trabalho apresenta uma análise de softwares que auxiliam no ensino e aprendizagem destes conceitos. Para tal, foram selecionadas cinco ferramentas (RoboMind, Scratch, SuperLogo, Kodu Lab e Alice), através de uma análise empírica sobre as interfaces, usabilidade e a fidelidade com os conceitos de computação, destacando os pontos positivos e negativos sob a ótica de licenciados em Computação.

### **3.3.2 Ensino de ciência da computação na educação básica: ferramentas, desafios e possibilidades**

**Autor(es):** Rozelma S. de França, Waldir C. da Silva, Haroldo J. C. do Amaral

**Fonte da Pesquisa:** Portal de Periódicos CAPES/MEC

**Ano de Publicação da Obra:** 2016

**Resumo/Abstract da Obra:** Este trabalho oferece contribuições acerca do ensino de Computação na educação básica no Brasil. Apresenta-se uma experiência de ensino de Computação numa escola da rede pública do Estado de Pernambuco com o emprego da Computação Unplugged e de uma linguagem de programação com abordagem lúdica.

### **3.3.3 Ferramentas de auxílio ao aprendizado de programação: um estudo comparativo**

**Autor(es):** Márcia Valéria Rocha de Souza, A. César C. França

**Fonte da Pesquisa:** Portal de Periódicos CAPES/MEC

**Ano de Publicação da Obra:** 2016

**Resumo/Abstract da Obra:** O objetivo deste artigo é refletir sobre a utilidade dos softwares de auxílio ao aprendizado de programação para alunos iniciantes. Para tanto, foi conduzido um levantamento dos conceitos de programação abordados em 9 ferramentas disponíveis na internet. Como resultado deste artigo, é sugerido um modelo de classificação em três níveis que permite a comparação e a adoção da ferramenta mais adequada de acordo com as características do estudante.

### **3.3.4 Learning and teaching programming: A review and Discussion**

**Autor(es):** Anthony Robins , Janet Rountree & Nathan Rountree

**Fonte da Pesquisa:** Taylor And Francis Online

**Ano de Publicação da Obra:** 2003

**Resumo/Abstract da Obra:** Neste artigo é feita uma revisão da literatura relacionada ao estudo psicológico / educacional da programação. Onde foram identificadas tendências gerais comparando programadores iniciantes e experientes, conhecimento de programação e estratégias.

### **3.3.5 Jogo sério colaborativo para o ensino da programação a crianças**

**Autor(es):** Admilo Ribeiro

**Fonte da Pesquisa:** Portal de Periódicos CAPES/MEC

**Ano de Publicação da Obra:** 2012

**Resumo/Abstract da Obra:** esta dissertação é sobre projetar, criar e testar um jogo sério para introduzir conceitos de programação às crianças chamado Boobo World.

### **3.3.6 Usando o SCRATCH para potencializar o pensamento criativo em crianças do ensino fundamental**

**Autor(es):** Amilton Rodrigo de Quadros Martins

**Fonte da Pesquisa:** Portal de Periódicos CAPES/MEC

**Ano de Publicação da Obra:** 2012

**Resumo/Abstract da Obra:** O presente trabalho tem como objetivo elucidar o potencial do uso de ambientes de programação de computadores no desenvolvimento do pensamento criativo em estudantes de ensino fundamental. A intenção foi buscar

subsídios para a compreensão do potencial desses ambientes no desenvolvimento do pensamento criativo, pautado na autonomia, na curiosidade e no protagonismo.

### **3.3.7 O uso de linguagem LOGO na educação infantil.**

**Autor(es):** Ronaldo Machado da Silva

**Fonte da Pesquisa:** Portal de Periódicos CAPES/MEC

**Ano de Publicação da Obra:** 2010

**Resumo/Abstract da Obra:** Este Artigo tem como objetivo expor o uso da linguagem de programação logo na educação de crianças, bem como quais são as vantagens deste uso na formação do intelecto da criança, ficando evidente que a Linguagem de Programação Logo é ótima para o desenvolvimento das habilidades intelectuais da criança, que ao interagir com o computador através da “tartaruga” constrói as estruturas mentais de forma privilegiada fazendo com que o raciocínio lógico fique bem mais desenvolvido favorecendo a aprendizagem das matérias curriculares principalmente a matemática.

### **3.3.8 Programar é bom para crianças? Uma visão crítica sobre o ensino de programação nas escolas.**

**Autor(es):** Wendell Bento Geraldês

**Fonte da Pesquisa:** Portal de Periódicos CAPES/MEC

**Ano de Publicação da Obra:** 2014

**Resumo/Abstract da Obra:** O artigo apresenta reflexões sobre o ensino de programação nas escolas e os impactos positivo e negativo dessa prática nos dias atuais. O texto trata ainda das iniciativas relativas ao ensino de programação nas escolas, considerando também a opinião de alguns especialistas sobre o assunto. Afinal, é bom para as crianças aprender a programar computadores nas escolas? Todas as pessoas podem aprender a programar computadores? Qual a importância da aprendizagem de programação para a sociedade atual? Em busca das respostas a essas questões, serão discutidas aqui as vantagens e desvantagens a respeito deste tema.

### **3.3.9 Programação de computadores como meio de desenvolvimento do raciocínio lógico em crianças e adolescentes.**

**Autor(es):** Juliane Colling, Carline Ternus, Beatriz Moesch, Silviane Lawall Soares, Marisley Vilas Bôas Soares.

**Fonte da Pesquisa:** Portal de Periódicos CAPES/MEC

**Ano de Publicação da Obra:** 2014

**Resumo/Abstract da Obra:** O tema deste estudo é o desenvolvimento do raciocínio lógico em crianças e adolescentes por meio da aplicação de aulas de programação de computadores. A problemática que se apresenta para este estudo é: “de que forma a realização de aulas de programação de computadores contribui para o processo de aprendizagem dos estudantes, principalmente no desenvolvimento da lógica de raciocínio? ”.

### **3.3.10 RoboEduc: especificação de um software educacional para ensino da robótica as crianças como uma ferramenta de inclusão digital.**

**Autor(es):** Viviane Gurgel de Castro

**Fonte da Pesquisa:** Portal de Periódicos CAPES/MEC

**Ano de Publicação da Obra:** 2008

**Resumo/Abstract da Obra:** Esta dissertação propõe um software educacional para o ensino da robótica denominado RoboEduc, criado para ser utilizado por crianças digitalmente excluídas do ensino fundamental. Sua implementação prioriza uma interface amigável, podendo ser ensinados conceitos de robótica e programação, de uma maneira inovadora, fácil e divertida. Com essa nova ferramenta, usuários sem conhecimento prévio algum de informática ou robótica são capazes de controlar um robô, previamente montado com os Kits Lego, ou até programa-lo para realizar determinadas tarefas.

### **3.3.11 Ensino de programação utilizando jogos digitais: uma revisão sistemática da literatura.**

**Autor(es):** Tainá Jesus Medeiros, Thiago Reis da Silva, Eduardo Henrique da Silva Aranha.

**Fonte da Pesquisa:** Portal de Periódicos CAPES/MEC

**Ano de Publicação da Obra:** 2013

**Resumo/Abstract da Obra:** Iniciativas internacionais renomadas indicam a importância do ensino de programação desde o ensino básico para formação nos profissionais de TI, mas existem ainda diversos desafios para a implantação desta ideia. Nesse contexto, este artigo tem como objetivo apresentar uma revisão



sistemática dos artigos referentes à utilização de jogos digitais para o ensino de programação, publicados nos últimos cinco anos (2008-2012) em três principais eventos nacionais na área de informática na educação e também em três relevantes revistas nacionais na área. Os resultados obtidos mostraram que as pesquisas focam principalmente no ensino de programação para o ensino médio e a maioria está sendo realizada por pesquisadores localizados nas em três regiões brasileira.

### **3.3.12 Avaliação empírica da utilização de um jogo para auxiliar a aprendizagem de programação.**

**Autor(es):** Elieser A. de Jesus, André L. A. Raabe

**Fonte da Pesquisa:** Portal de Periódicos CAPES/MEC

**Ano de Publicação da Obra:**

**Resumo/Abstract da Obra:** Este artigo descreve uma avaliação quantitativa sobre o efeito da utilização de um jogo de computador como ferramenta de apoio à aprendizagem de programação. Neste jogo os alunos precisavam programar para vencer. Foram realizados experimentos com 3 turmas em um delineamento experimental com pré e pós-teste e grupo de controle. Não foi observada nenhuma diferença significativa no progresso dos alunos que jogaram em comparação com os que não jogaram (o grupo de controle). Outros resultados bem como a validade do experimento são discutidos em maiores detalhes no artigo.

### **3.3.13 Modelação de sistemas experimentais em Toon Talk para ambientes educativos: ciclo da água.**

**Autor(es):** Luís Manuel Mesquita dos Santos

**Fonte da Pesquisa:** Repositório da Universidade de Trás-os-Montes e Alto Douro

**Ano de Publicação da Obra:** 2009

**Resumo/Abstract da Obra:** Esse trabalho consistiu na realização de um estudo com seis alunos do ensino médio, com o intuito de aplicar em ambiente educativo a ferramenta de programação ToonTalk, tendo a temática da água como matéria basilar; como objeto de estudo, foram também observadas as atitudes e reações destes alunos enquanto manipulavam este programa. Esta investigação assumiu um carácter qualitativo, descrevendo e interpretando as observações do grupo de alunos escolhidos, enquanto caso de estudo.

### **3.3.14 Limitações da utilização do Alice no ensino de programação para alunos de graduação.**

**Autor(es):** Joselaine Valaski, Emerson Cabrera Paraiso

**Fonte da Pesquisa:** Portal de Periódicos CAPES/MEC

**Ano de Publicação da Obra:** 2012

**Resumo/Abstract da Obra:** Este trabalho avaliou o uso do Alice em uma turma do curso de graduação de Sistemas de Informação. Os resultados mostraram que, diferentemente de relatos anteriores, o uso do Alice não contribuiu de forma significativa na aprovação da disciplina Programação I. Os motivos que os fez acreditar que este tipo de ambiente tem aplicação limitada para alunos dos cursos de programação em nível de graduação são discutidos neste trabalho.

### **3.3.15 Lowering the Barriers to Programming: a survey of programming environments and languages for novice programmers.**

**Autor(es):** Caitlin Kelleher, Randy Pausch

**Fonte da Pesquisa:** Repository School of Computer Science, Carnegie Mellon University

**Ano de Publicação da Obra:** 2003

**Resumo/Abstract da Obra:** Este artigo apresenta uma taxonomia de linguagens e ambientes projetados para torna mais acessível a programação para iniciantes de todas as idades. Os sistemas são organizados pelo seu principal objetivo, seja para ensinar programação ou para usar a programação para capacitar seus usuários.

### **3.3.16 Avaliação empírica da utilização de um jogo para auxiliar a aprendizagem de programação.**

**Autor(es):** Débora Pereira Coura

**Fonte da Pesquisa:** Portal de Periódicos CAPES/MEC

**Ano de Publicação da Obra:** 2006

**Resumo/Abstract da Obra:** Essa dissertação apresenta um estudo sobre os principais sistemas que trabalham com técnicas de programação por demonstração. Levando em consideração os pontos fracos dos sistemas estudados e propondo melhorias para dos mesmos como: a utilização de regras escritas em primeira pessoa, o uso da herança e o uso de regras com condições negativas.

### **3.3.17 Relato de experiência da utilização da ferramenta *Lightbot* enquanto construtor do raciocínio para lógica de programação.**

**Autor(es):** Mariane Regina Sponchiado Cassenote, Rodrigo Luiz Antoniazzi

**Fonte da Pesquisa:** Portal de Periódicos CAPES/MEC

**Ano de Publicação da Obra:** 2015

**Resumo/Abstract da Obra:** É um projeto de extensão que visa capacitar estudantes da rede pública de ensino na utilização de softwares ou ferramentas apropriadas para desenvolver o raciocínio lógico voltado à programação de computadores, assim como a capacidade de simplificar problemas complexos.

### **3.3.18 LogicBlocks: Uma ferramenta para o ensino de lógica de programação.**

**Autor(es):** Rogerio Paulo Marcon Junior, Bruno Batista Boniati

**Fonte da Pesquisa:** Portal de Periódicos CAPES/MEC

**Ano de Publicação da Obra:** 2015

**Resumo/Abstract da Obra:** Este trabalho apresenta a ferramenta LogicBlocks, um website desenvolvido para propor e resolver desafios de lógica de programação através de blocos lógicos de encaixe utilizando a API Blockly.

### **3.3.19 Análise da ferramenta de programação visual *Blockly* como recurso educacional no ensino de programação.**

**Autor(es):** Garibaldi da Silveira Júnior, Fábio Diniz Rossi, Patric Lincoln Ramires Izolan, Jader Renan da Silva Almeida

**Fonte da Pesquisa:** Portal de Periódicos CAPES/MEC

**Ano de Publicação da Obra:** 2014

**Resumo/Abstract da Obra:** Este artigo visa analisar a ferramenta de programação visual Blockly, como uma alternativa para que alunos do ensino médio e fundamental conheçam conceitos de algoritmos e desenvolva o raciocínio Lógico.

### **3.3.20 MODEBOTS: Environment for Programming Robots for Children Between the Ages of 4 and 6.**

**Autor(es):** Kryscia Ramírez-Benavides, Luis A. Guerrero

**Fonte da Pesquisa:** IEEE Xplore

**Ano de Publicação da Obra:** 2014

**Resumo/Abstract da Obra:** O projeto apresentado neste artigo descreve protótipos para o primeiro contato com um ambiente de programação de crianças de 4 a 6 anos. Propondo um novo processo de aprendizagem em crianças pré-escolares.

### **3.3.21 Um estudo exploratório dos games para introdução ao pensamento computacional.**

**Autor(es):** Sérgio Souza Costa, Spartacus Silva Souza, Leonardo C. C. Mendes, Rosane de F. A. Obregon, Luzia Emanuelle R. V. da Silva, Evaldinolia Gilbertoni Moreira, Jeane Silva Ferreira

**Fonte da Pesquisa:** Portal de Periódicos CAPES/MEC

**Ano de Publicação da Obra:** 2015

**Resumo/Abstract da Obra:** este artigo objetiva identificar e analisar os games empregados para o ensino e aprendizado do pensamento computacional. O estudo levou em consideração algumas dimensões, como o gênero, habilidades computacionais exploradas e a linguagem utilizada.

## **4 LINGUAGENS E AMBIENTES DE ENSINO DE PROGRAMAÇÃO PARA CRIANÇAS**

Com base na leitura dos artigos selecionados na revisão sistemática, foi desenvolvido um método de classificação das muitas linguagens e ferramentas de programação que tentam tornar mais acessível o aprendizado de programação para iniciantes, e também introduzir conceitos de programação num nível mais baixo para que crianças tenham acesso e mais facilidade no uso delas.

Todas as ferramentas desenvolvidas para essa finalidade têm por objetivo tornar o aprendizado de uma linguagem ou a lógica de uma linguagem mais fácil. Isso só pode ser possível se esses ambientes forem acessíveis para as crianças. Segundo Ribeiro (2012; apud Papert, 2012) está linguagem ou ambiente de ensino de programação para crianças tem que respeitar os seguintes princípios:

- **“Low floor”** – Fácil de começar, ser inicialmente interessante.
- **“High Ceiling”** – Que projetos mais complexos possam ser incrementados com o passar do tempo.

- “**Wide Walls**” – capacidade de motivar diferentes públicos, de forma que pessoas com interesses diferentes se sintam motivadas.

Esses sistemas tem o objetivo de facilitar e ajudar no aprendizado da essência da programação. No entanto esses sistemas têm dois objetivos opostos quando estão sendo criados: um sistema de fácil uso inicialmente e que possa ser atrativo à primeira vista, ao mesmo tempo que precisa fornecer uma base de conhecimentos para o usuário como suporte para linguagem de programação mais avançadas. Dito isso, a ferramenta precisa ter um equilíbrio entre esses dois pontos conflitantes para que possa solucionar o problema de uma criança que está iniciando em um ambiente de programação (KELLEHER e PAUSCH, 2003).

#### 4.1 METODOLOGIA DOS SISTEMAS DE ENSINO DE PROGRAMAÇÃO

A primeira metodologia desenvolvida especificamente para o de ensino de programação para crianças foi o “construcionismo”, proposto e desenvolvido por Seymour Papert e utilizada para a interação aluno-objeto na linguagem também desenvolvida por Papert chamada “LOGO”, que já foi descrita anteriormente neste trabalho. Essa metodologia foi usada em várias aplicações que vieram depois do LOGO, e tem grande influência nas ferramentas usadas atualmente. Esses sistemas além de usarem o construcionismo que se baseia na falha como processo de aprendizagem. Usam outros mecanismos de programação que serão descritos mais detalhadamente neste capítulo.

De acordo com KELLEHER e PAUSCH (2003) esses sistemas de ensino de mecanismos de programação têm objetivos específicos que são: Ensinar a criar, ensinar a estruturar e compreender a execução, como pode ser visto na figura 4.1.

**Figura 4.1:** Mecanismos de Ensino de Programação

Fonte: Ribeiro (2012)

#### 4.1.1 Criação, Estruturação e Execução

Para se programar é preciso inserir linhas de código num editor com instruções do que você quer que esse programa faça. Há um problema de se conseguir traduzir suas intenções para quem está iniciando em uma linguagem de programação. Para resolver esse problema existem meios alternativos que se tornam mais atraentes tanto para crianças quanto para quem está iniciando. Alguns exemplos desses meios são a substituição do código por objetos gráficos, representando um comando, estrutura ou variável. Utilização dos resultados das ações na interface para criar um programa, e múltiplos métodos para criação dos programas, por exemplo manipulação direta ou preenchimento de formulários (RIBEIRO, 2012).

Já na fase de estruturação, as Ferramentas não se preocupam com a sintaxe dos códigos e sim com a estruturação e organização deles. Na etapa de execução as ferramentas pretendem ensinar aos utilizadores iniciantes a perceber a execução dos seus programas, uma vez que nem sempre o que se escreve é o que o utilizador quer fazer, aprendendo assim a compreender os erros do programa (KELLEHER e PAUSCH, 2003).

## 4.2 CARACTERÍSTICAS DAS LINGUAGENS E AMBIENTES DE PROGRAMAÇÃO

Nesta seção serão apresentadas ferramentas de apoio a aprendizagem de programação para crianças e suas principais características. Como existe um número imenso de ferramentas com essa finalidade foram analisadas as mais usadas e populares, e também as que tinham trabalhos em comum selecionados na etapa da revisão sistemática. Essencialmente todos os sistemas de ensino de programação tem que ter os três requisitos dispostos no tópico 4.1.1.

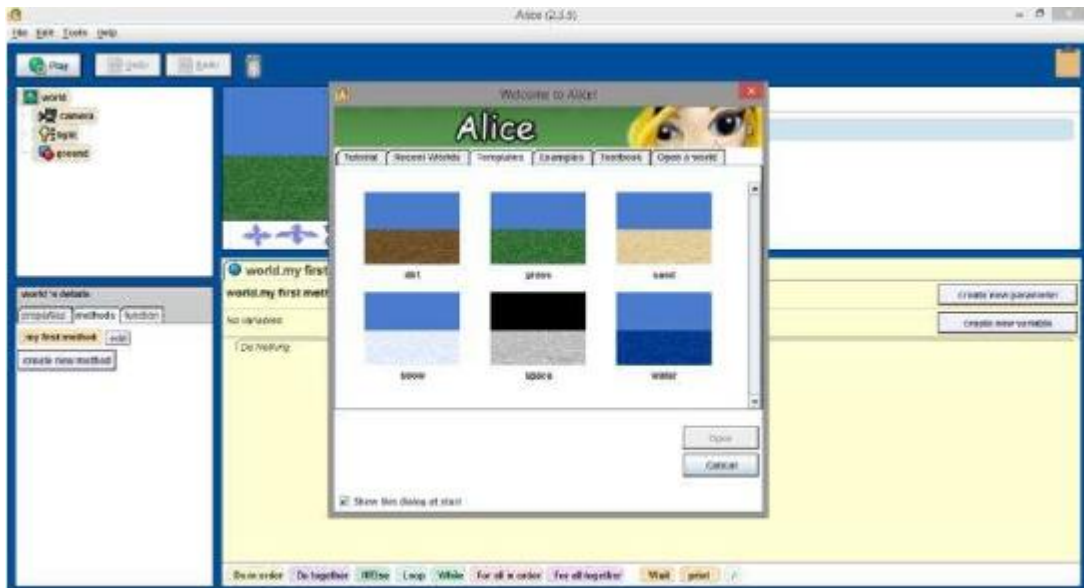
### **Alice**

É um ambiente de programação tridimensional, como pode ser visto na figura 4.2, que foi desenvolvido na Universidade de Virginia em conjunto com a Universidade de Carnegie Mellon, por um grupo liderado pelo professor de ciência da computação da Carnegie Mellon Randy Pausch (Hebert 2011, apud SILVA et al, 2014). Alice é uma ferramenta destinada a iniciantes, cujo seu principal objetivo é fazer uma primeira exposição à programação orientada a objetos, permitindo aos alunos conhecer conceitos fundamentais no contexto de criação de vídeos animados ou jogos simples (COOPER E DANN 2000, apud RIBEIRO, 2012).

O ambiente virtual do Alice é denominado world onde é feito a partir de elementos pré-programados as interações. A partir desses elementos pré-programados é possível criar histórias seguindo uma construção lógica, ver figura 4.2, os usuários podem arrastar gráficos, onde existem instruções correspondentes a comandos padrões na produção de programação orientada a objetos, semelhante ao Java e C++ (BARROS et al, 2012 apud SILVA et al, 2014).

Os objetos 3D presentes em Alice (pessoas, animais e veículos) povoam um mundo virtual onde os estudantes podem criar programas para os animar. Sendo esses programas scripts simples, que durante a sua execução permitem aos objetos receber os inputs de quem estar utilizando via mouse ou teclado e depois disso, cada ação é animada durante o período de tempo que foi especificado pelo utilizador (COOPER E DANN 2000, apud RIBEIRO, 2012).

**Figura 4.2:** Interface Inicial do Alice



Fonte: Ribeiro (2012)

Como linguagem de programação, as principais características de Alice são (COOPER E DANN 2000, apud RIBEIRO, 2012):

- **Ações** – São transformadores de estado, Alice fornece um conjunto de ações que podem ser divididas em duas categorias: os que dizem ao objeto que movimento realizar e aqueles que mudam a aparência do objeto.
- **Procedimentos** – Alice permite sequencias de instruções, conceito bem similar em muitas outras linguagens de programação.
- **Funções** – Desempenham papéis distintos: implementação de recursividade, implementação das interações via eventos.
- **Decisões** – As decisões correspondem aos comandos das ações que os objetos devem efetuar.
- **Recursividade** – Alice fornece suporte a repetições por um comando de instrução de Loop.
- **Eventos/Interações** – Suporta handlers para eventos e criação de interfaces gráficas (GUI).

Atualmente o Alice está na sua versão 3.2, mas a versão 2.4 ainda está disponível no site oficial Alice.org para os sistemas operacionais Windows, MAC OS X e Linux. A primeira versão foi disponibilizada em 1999 e passou por muitas atualizações O Alice também serve também como base para outra linguagem e ambiente de programação **designado MAMA** (KELLEHER e PAUSCH, 2003).

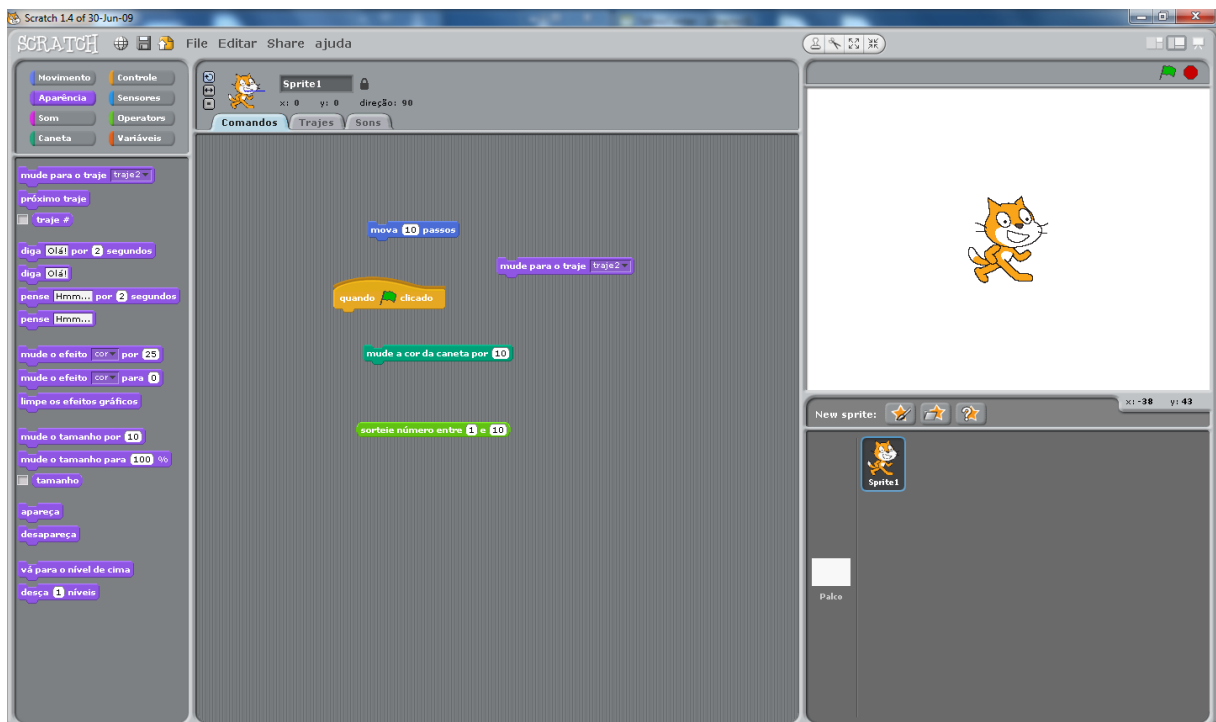


## Scratch

O Scratch é um ambiente de desenvolvimento visual criado por Lifelong Kindergarten Group (LLK), grupo de pesquisa do MIT Media Lab. O ambiente tem entre seus objetivos o propósito de introduzir a programação de maneira fácil e rápida para quem não possui nenhum tipo de experiência em programação. O Scratch utiliza uma linguagem visual, onde é possível criar jogos, animações e histórias interativas (MALONEY et al. 2010 apud SILVA et al, 2014).

O scratch utiliza uma linguagem visual, ver figura 4.3, que permite que utilizadores trabalhem em projetos de seu interesse como animações ou jogos, e aprendam programação ao mesmo tempo. O alvo no desenvolvimento dessa ferramenta eram as crianças entre 8 e 16 anos de idade, mas o scratch tem como principal objetivo introduzir a programação para quem nunca teve contanto com ela.

**Figura 4.3:** Tela Principal do Scratch



Fonte: Ribeiro (2012)

Segundo Ribeiro (2012; apud MALONEY et al, 2010), o Scratch pode ser analisado de duas formas:

- **Ambiente de Programação:** O ambiente do Scratch pretende encorajar a autoaprendizagem, fazendo o utilizador experimentar o uso de seus scripts e recebendo um feedback ao mesmo tempo. O Scratch possui uma interface

baseada na filosofia *Single-Window User Interface* que permite que todos os componentes chave do Scratch estejam sempre visíveis.

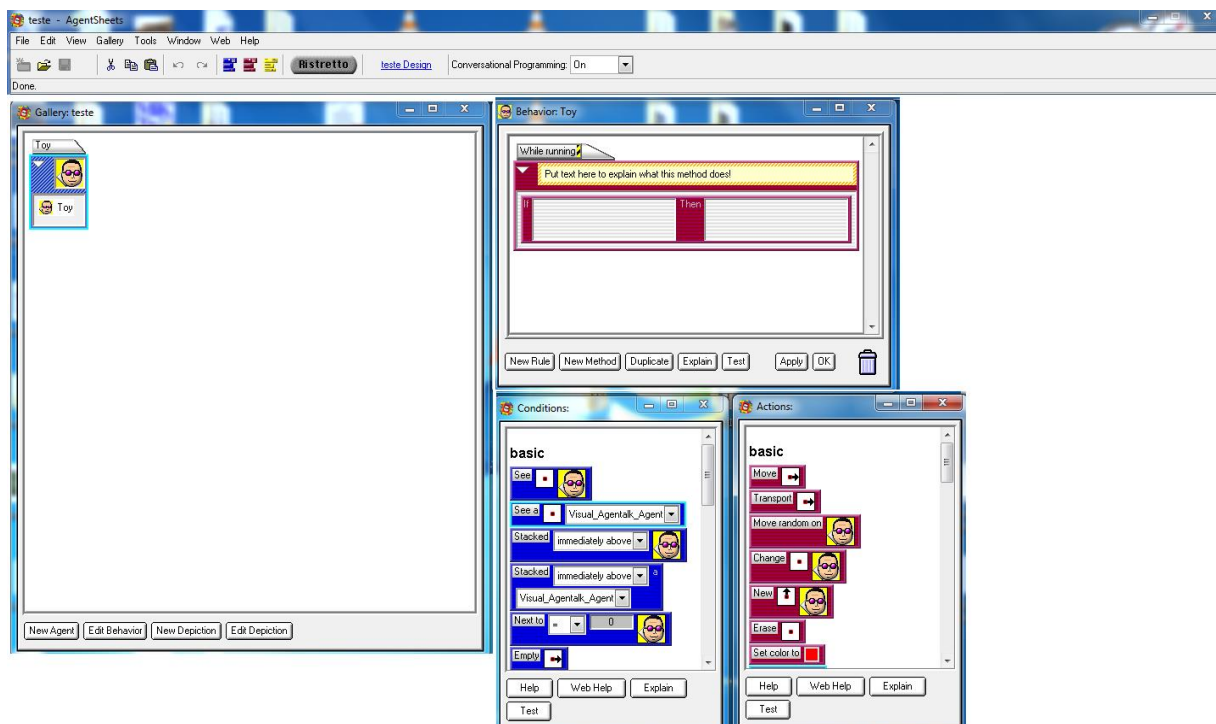
- **Linguagem de programação:** A programação é feita juntando blocos coloridos, com instruções para controlar objetos gráficos em 2D, chamados de sprites. O conjunto de blocos empilhados constituem os scripts que representam declarações, expressões e estruturas de controle. Cada bloco tem uma forma própria ajudando assim o utilizador a encaixa-los corretamente evitando scripts que não façam sentido.

## AgentSheets

Assim como no scratch, o AgentSheets também usa a filosofia *drag-and-drop*, como pode ser visto na figura 4.4, fazendo com que seja mais fácil criar simulações interativas, explorar novas ideias, testar teorias e explorar processos complexos no campo da ciência (INC. AGENTSHEETS, 2011 apud RIBEIRO, 2012).

As simulações nessa ferramenta são baseadas em agentes e depois podem ser publicadas na Web, as simulações são feitas especificando o comportamento dos sprites sobre um mundo representado por uma grelha. Esses sprites podem ser movidos para novas posições, produzir sons ou mudar de sua aparência (KELLEHER e PAUSCH, 2003).

Figura 4.4: Tela do AgentSheets



Fonte: Ribeiro (2012)

## Kodu

É uma linguagem visual para criação de jogos, destinada a crianças desenvolvido pela Microsoft, que pode ser executado no Windows e no Xbox 360. O Kodu usa uma interface que usa uma linguagem simples e totalmente baseada em ícones. Figura 4.5. (RIBEIRO, 2012)

A ferramenta Kodu permite a programação de cada personagem e objeto individualmente para interagir com o mundo, podendo funcionar como agentes inteligentes. Os programas são expressos em termos físicos, usando conceitos como visão, audição, e tempo para controlar o comportamento do personagem (MICROSOFT RESEARCH, 2011 apud RIBEIRO, 2012).

**Figura 4.5:** Tela do Kodu



**Fonte:** Silva et al (2014)

Já na parte da programação, os usuários têm a opção de programar seus próprios personagens, apesar de existir uma opção a qual o usuário pode inserir seus personagens já pré-programados para executarem certos tipos de tarefas. O principal ponto positivo do Kodu é a maneira lúdica e interativa em que são apresentados os conceitos de lógica para as crianças. Dentre suas limitações podem ser observadas a quantidade escassa de personagens, há apenas robôs, discos voadores, naves e outros “figurantes” semelhantes, e a quantidade de objetos que podem ser inseridos dentro do mundo construído também pode ser citado como outro problema desta ferramenta (SILVA et al, 2014).

## Squeak Etoys

É um ambiente, como pode ser visto na figura 4.6, de programação orientada a objetos para o uso na educação de crianças, a linguagem Squeak foi desenvolvida originalmente na Apple. O Squeak Etoys suporta gráficos 2D e 3D, imagens, texto, apresentações, páginas Web, vídeos, ...etc (RIBEIRO, 2012).

O Squeak Etoys foi a principal influência de outro ambiente de programação voltado ao ensino de crianças, o scratch que foi projetado com códigos do Etoys no começo do século XXI. Uma das características do Etoys é a partilha em tempo real dos projetos criados por outros utilizadores do sistema (KAY, 2005 apud RIBEIRO, 2012).

Figura 4.6: Tela Inicial do Squeak Etoys



Fonte: Ribeiro (2012)

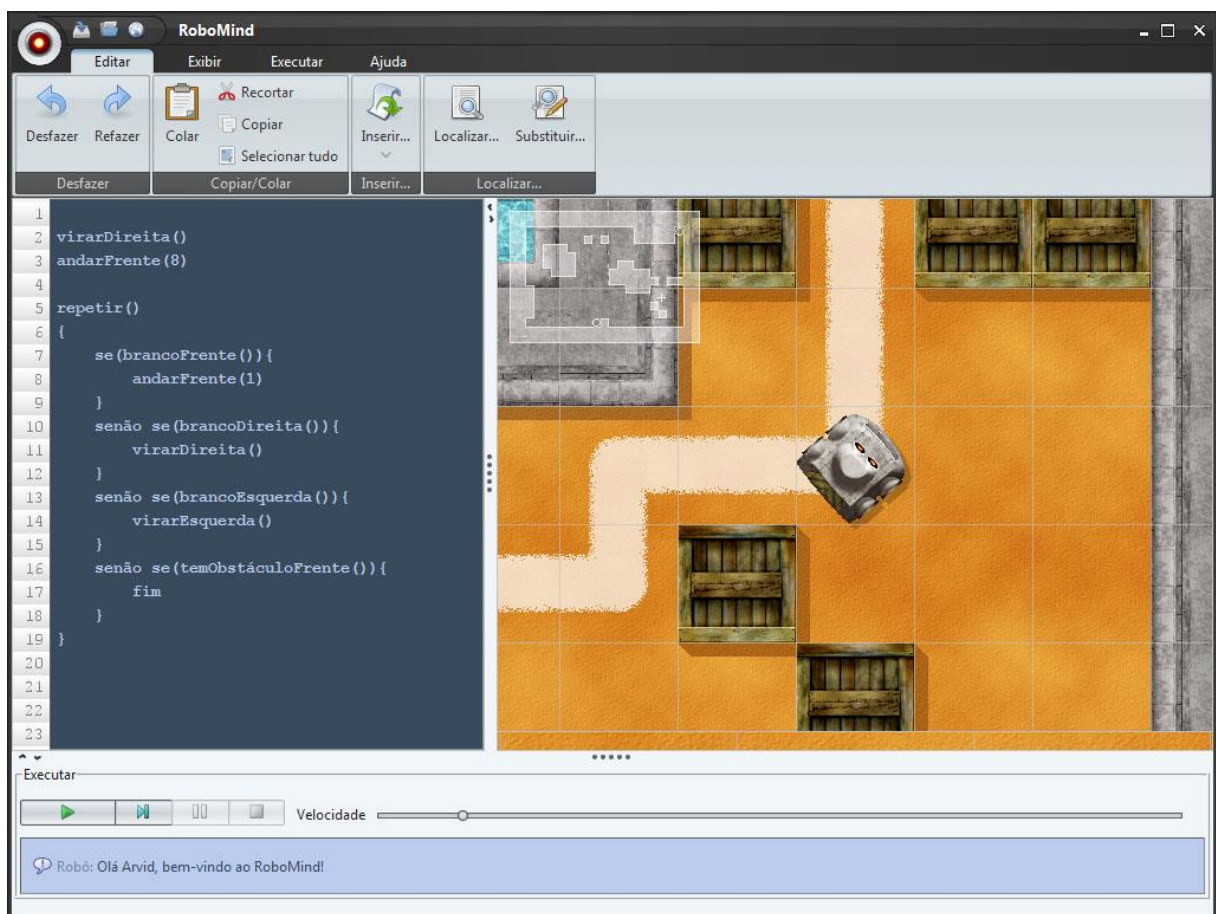
## RoboMind

É um ambiente desenvolvido na universidade de Amsterdam, sua tela inicial pode ser vista na figura 4.7, com o objetivo de facilitar o aprendizado de conceitos iniciais de programação. Nesse ambiente é possível realizar um ensino de conceitos de lógica e ter uma noção inicial nas áreas de inteligência artificial e robótica. A

linguagem utilizada no RoboMind é chamada Robo/Roo, onde é possível programar um robô para realizar movimentos dentro de um mapa bidimensional (SILVA et al, 2014).

A linguagem contempla comandos para estrutura de seleção, e repetição. Onde através desses principais comandos da linguagem, torna-se possível adotar o RoboMind para o ensino de conceitos de lógica para programação associado aos conceitos de expressões lógicas, estruturas de decisão e estruturas de repetição (SILVA et al, 2014).

**Figura 4.7:** Tela do RoboMind



**Fonte:** Silva et al (2014)

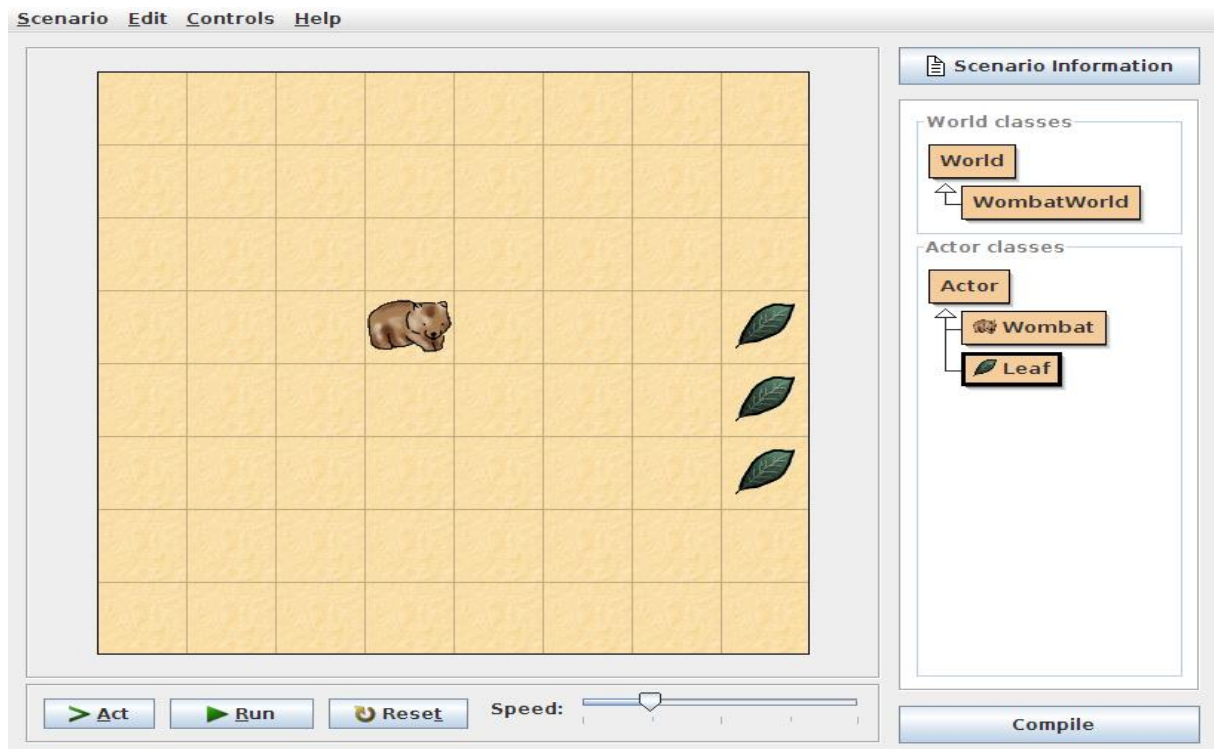
Segundo (VAHLICK et al, 2009 apud SILVA et al, 2014) usando o RoboMind no ensino de programação, os alunos têm contato de uma maneira mais simples ao estudo de algoritmos e podem executa-los por inteiro ou passo-a-passo, criando assim uma compreensão melhor das estruturas de programação.

## Greenfoot

É um ambiente interativo baseado em Java, figura 4.8, ele permite o fácil desenvolvimento de aplicativos gráficos bidimensionais, como simulações e jogos interativos. O Greenfoot é um software livre lançando com o apoio da Oracle que está disponível para Windows, OS X, Linux, Solaris e qualquer JVM recente. O sistema do Greenfoot é composto por duas partes: sendo a primeira, um ambiente integrado de desenvolvimento pedagógico que integra tanto ferramentas comuns (como um editor, compilador, máquina virtual), como ferramentas educacionais (como a invocação de objetos interativos, a inspeção, a visualização de classes). E a segunda parte, um framework de simulação de um micromundo (KOLLING, 2008 apud RIBEIRO, 2012).

O ambiente de Greenfoot fornece uma plataforma que permite uma grande interação visual a introdução da programação orientada a objetos em Java. Os estudantes que utilizam este sistema concentram a sua atenção na programação do comportamento do objeto. O código em Greenfoot é apresentado de forma gráfica, desta forma os estudantes focam-se nos conceitos importantes em vez de preocuparem-se com a sintaxe da linguagem (RIBEIRO, 2012).

**Figura 4.8:** Tela Inicial do Greenfoot



Fonte: Ribeiro (2012)

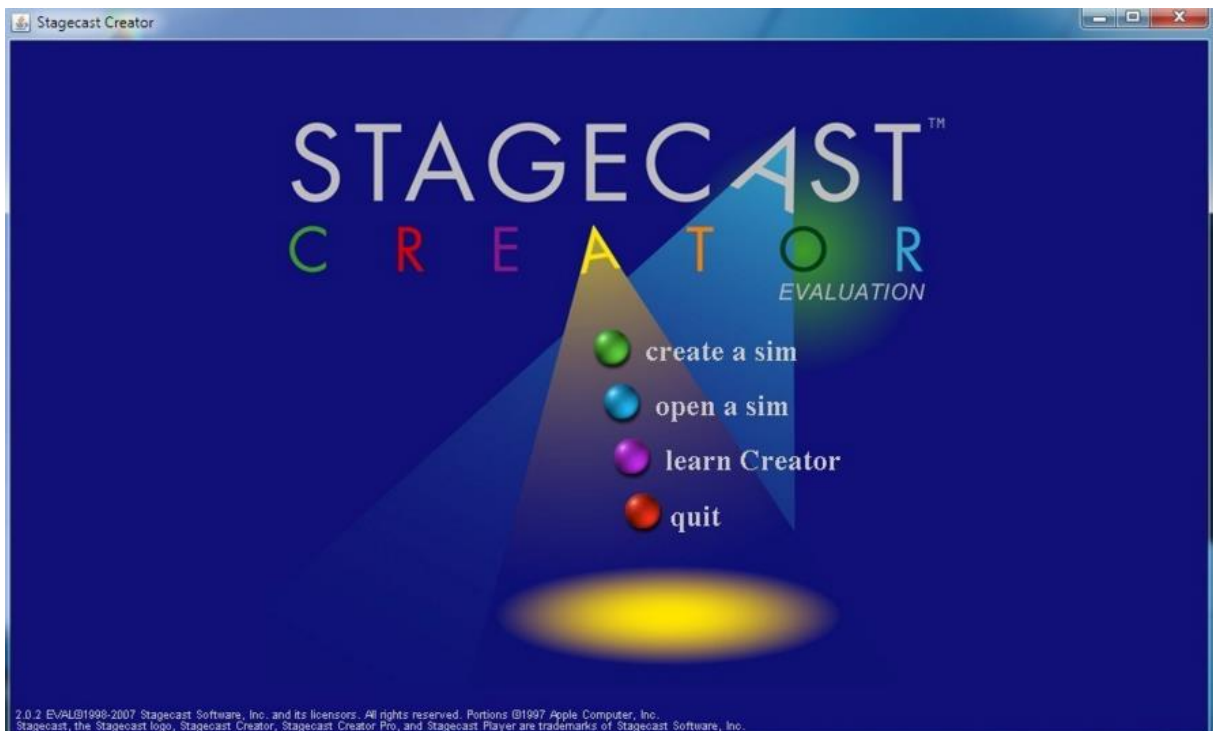
## Stagecast Creator

O Stagecast Creator é baseado no conceito de programação por demonstração, figura 4.9, onde as regras são criadas, dando exemplos de ações que devem ocorrer em uma dada situação. Ele pode ser usado para a construção de simulações, animações e jogos, que são executados em qualquer plataforma Java adequada. (KELLEHER e PAUSCH, 2003)

Desenvolvido em 1999, o Stagecast Creator foi um dos primeiros produtos comerciais a utilizarem a chamadas técnicas de PBD (programação por demonstração), são ambientes que usam programação intuitivas, fáceis de serem aprendidas e especialmente desenvolvidas para não programadores criarem seus próprios jogos e simulações. Similares ao Stagecast Creator existem outras ferramentas que usam essas técnicas de PBD que são o AgentSheets e o Toon Talk (COURA, 2006).

Usando a técnica de PBD, o programa é escrito na linguagem do domínio do problema, sem utilizar termos de linguagens de programação convencionais. O usuário ganha em comodidade e simplicidade. É uma forma bem mais fácil de programar bem mais intuitiva (COURA, 2006).

**Figura 4.9:** Tela Inicial do Stagecast Creator



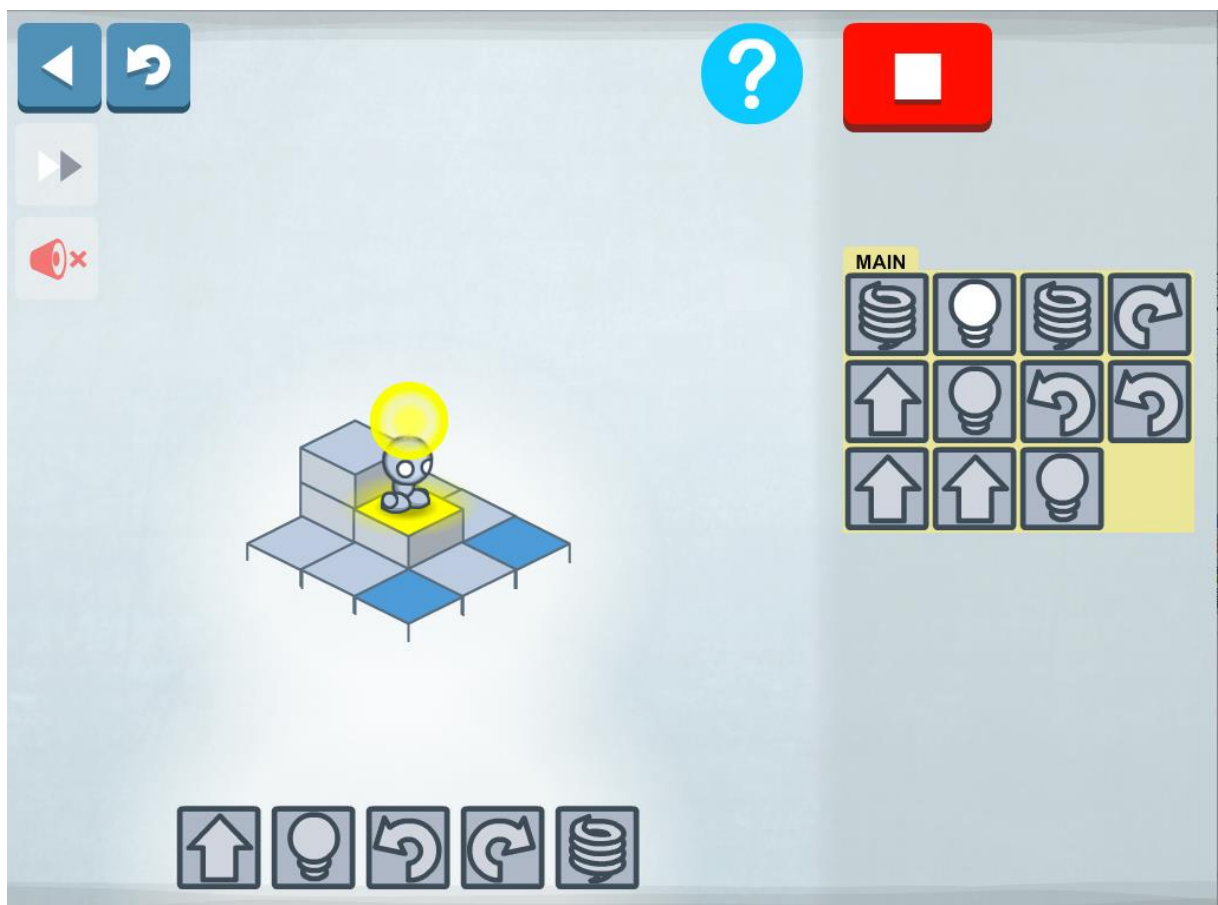
Fonte: Coura (2006)

## Lightbot

É um jogo online executado com o auxílio de uma plataforma Flash, como pode ser visto na figura 4.10, e pode ser acessada de qualquer navegador. A ferramenta é grátis e indicada para crianças que querem ter o primeiro contato com a lógica de programação.

Desenvolvido para diversas plataformas, ele utiliza cinco comandos para movimentar um robô em um espaço virtual. Os comandos são: seguir em frente, virar à esquerda, virar à direita, pular e acender uma luz. Estes comandos são representados como ícones e são arrastados para montar o algoritmo. Através do alcance do objetivo, mais funções vão liberadas. O objetivo do jogo é movimentar um robô dentro de um espaço quadriculado iluminando os quadrados que são solicitados. Além de ter suporte a procedimentos e recursividade (MARANHÃO, 2015).

**Figura 4.10:** Tela do Lightbot



Fonte: Lightbot (2016)

## Blockly

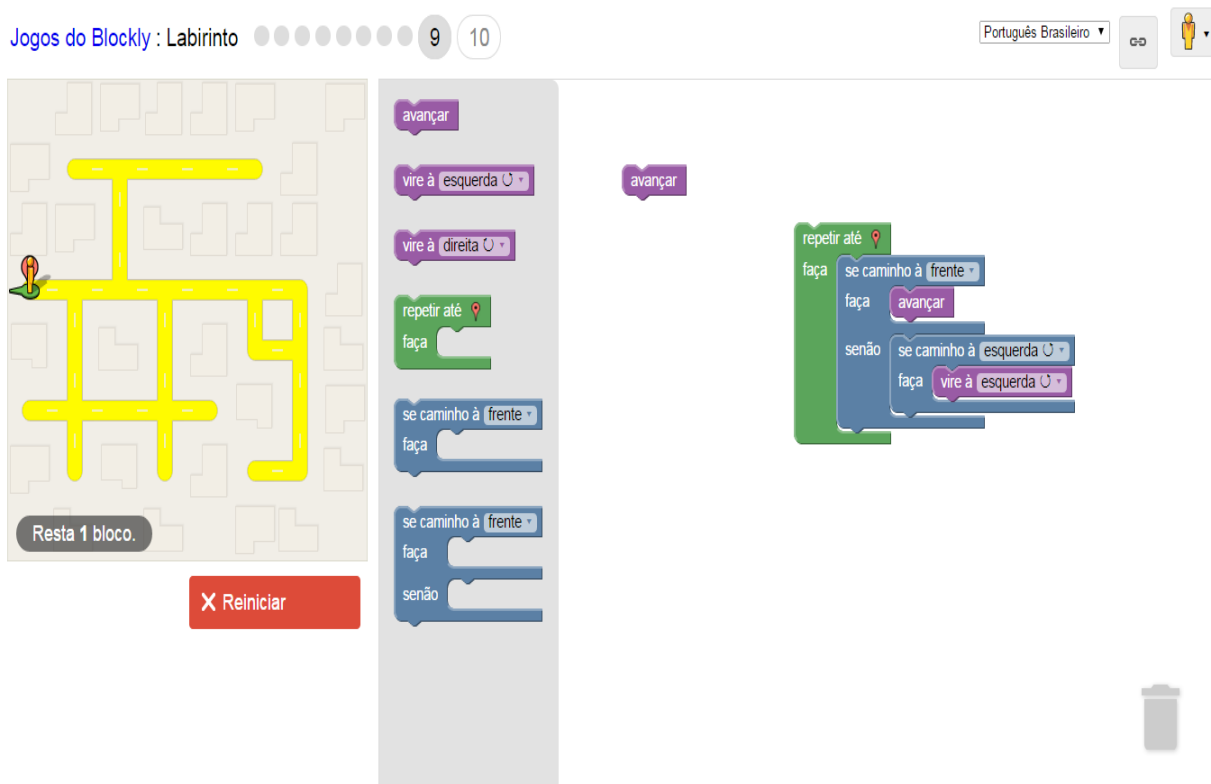
É uma linguagem desenvolvida pela Google, ver figura 4.11, segundo Silveira et al (2014; apud Marron, 2012) o Blockly segue os princípios e os formatos



apresentados no Scratch. A principal diferença é que a ferramenta é capaz de transformar o código criado pelos blocos em um código funcional em Javascript.

A partir da conexão dos blocos de encaixe com funcionalidades pré-definidas, o usuário pode escrever um pequeno roteiro de um programa e posteriormente executá-lo ou transformá-lo no respectivo código (são suportadas as linguagens de programação PHP, Python, Dart e JavaScript). Com o Blockly é mais difícil cometer um erro de sintaxe ou mesmo ambiguidades devido ao uso inadequado de parênteses (JÚNIOR E BONIATI, 2015).

**Figura 4.11:** Tela do Blockly



**Fonte:** Blockly (2016)

## Kodable

É uma aplicação lançada em 2012 para o iPad e Web, ver figura 4.12, sendo a versão para internet grátis. É indicado para o ensino de programação a crianças de 4 a 11. Segundo Ramirez-Benavides (2015) as crianças que o utilizam podem desenvolver habilidades para resolução de problemas, habilidades de comunicação e colaboração e o pensamento crítico, além de paradigmas computacionais como variáveis, classes, operadores lógicos etc. Sendo uma linguagem de programação baseada a ícones, o Kodable tenta ensinar os conceitos fundamentais de lógica e

programação, mediante comandos de arrastar e levar um personagem de um ponto inicial para um ponto final.

**Figura 4.12:** Kodable Sendo Executado



**Fonte:** Kodable (2016)

## Code.org

É um website desenvolvido com o intuito de incentivar o ensino de programação dentro de ambientes de ensino básico, como pode ser visto na figura 4.13. Para que isso seja possível, a aplicação apresenta diferentes tipos de jogos baseados em lógica e blocos, tendo como forma de chamar a atenção, a utilização de personagens infantis conhecidos como participantes dos problemas a serem resolvidos. Em 9 de dezembro de 2013, o Code.org lançou a iniciativa “Hour of Code 2013”, em português, “A hora do código 2013”, em que qualquer pessoa poderia dedicar uma hora para aprender a programação (JÚNIOR E BONIATI 2015, GERALDES 2014).

Criada em 2011 pelo iraniano Hadi Partovi, formado em ciências da computação pela universidade de Harvard, ele teve a ideia de criar um vídeo para estimular jovens a aprender programação usando depoimentos de personalidades do mundo da tecnologia. Com a ajuda de seu irmão lançou em 2013 o projeto para promover o ensino de programação nas escolas. No início o Code.org recebeu o apoio de 60 pessoas, entre elas Bill Gates, Mark Zuckerberg.

**Figura 4.13:** Ambiente Code.org

Fonte: Geraldles (2014)

Primeiro há um tutorial para iniciantes, onde aluno tem o primeiro contato com a programação através da linguagem Blockly, que possibilita arrastar e soltar blocos para escrever os códigos. O objetivo do primeiro programa é conseguir que o personagem Angry Bird atravesse um labirinto para chegar ao Porco Verde. O lado esquerdo do labirinto apresenta a área onde o programa é executado. Abaixo desse labirinto pode-se encontrar as instruções para cada quebra-cabeça. No meio há uma caixa de ferramentas que possui os comandos necessários para movimentar o personagem. À direita fica o local chamado de área de trabalho, para onde o aluno arrastará os blocos da caixa de ferramentas para construir o programa. Para apagar os blocos, é possível arrastá-los para uma lixeira que fica no canto da tela (GERALDES, 2014).

### 4.3 ANÁLISE DAS FERRAMENTAS

Após o levantamento das ferramentas que seriam analisadas, foi feito um estudo sobre como poderia ser feita a classificação delas. Foi conduzido uma verificação de conceitos básicos de programação abordados pelas ferramentas tentando achar pontos em comum entre elas e que tipo de referência cada ferramenta

tomou antes de ser implementada. Tomando como base a análise de outros trabalhos que foram selecionados na revisão sistemática.

Segundo Coura (2006) todas as ferramentas voltadas a resolução do problema de ensinar programação a iniciantes sofre algum tipo de influência de ferramentas de vieram anteriormente, a partir dessa analise podemos definir um método de comparação entre elas e aspectos que podem ser discutidos como: estilo de programação, estruturas de programação suportadas, como essas ferramentas foram construídas e a maneira que esses sistemas tentaram tornar a programação mais acessível para não programadores. Antes das características técnicas das ferramentas foram analisadas suas características gerais, mostradas na tabela 4.1.

**Tabela 4.1:** Características Gerais das Ferramentas Avaliadas

	Idioma	Licença
<b>Alice</b>	Inglês	Freeware
<b>Scratch</b>	40 idiomas	Código aberto com componentes proprietários
<b>AgentSheets</b>	Inglês	Software Proprietário
<b>Kodu</b>	24 idiomas	Freeware para Windows
<b>Squeak Etoys</b>	Inglês	Open Source
<b>RoboMind</b>	22 idiomas	Software Proprietário
<b>Greenfoot</b>	18 idiomas	Freeware
<b>Stagecast Creator</b>	Inglês	Freeware
<b>Lightbot</b>	Inglês	Disponível online
<b>Blockly</b>	40 idiomas	Disponível online
<b>Kodable</b>	Inglês	Disponível online
<b>Code.org</b>	40 idiomas	Disponível online

Nesta análise inicial podemos destacar que ferramentas feitas para o mercado tem mais opção de idiomas para os usuários como é o caso do Kodu mantido e desenvolvido pela Microsoft, e os de licença privada como RoboMind e o Scratch, que apesar de ser código aberto tem algumas restrições do proprietário. Softwares freeware foram em sua maioria desenvolvidos academicamente como é o caso do

Greenfoot desenvolvido e mantido pela Universidade de Kent, no Reino Unido. Uma exceção é o Squeak Etoys que foi originalmente desenvolvido pela Apple sobre o domínio “Squeak License” e foi novamente licenciado em 2006, e atualmente é inteiramente livre e de código aberto. Já as ferramentas web algumas tem várias opções de idioma como é o caso do Code.org, apesar da tradução deixar a desejar em alguns pontos da ferramenta.

#### 4.3.1 Conceitos Avaliados das Ferramentas

Para a avaliação das ferramentas foram considerados conceitos de programação como paradigmas de programação, estruturas suportadas pelas ferramentas, como o código é representado, como acontece a construção dos programas e métodos de ensino buscando facilitar o entendimento de crianças e usuários iniciantes no estudo da lógica e de uma linguagem de programação.

Um paradigma de programação fornece e determina a visão que o usuário tem sobre a estruturação e execução do programa. As ferramentas levantadas serão avaliadas sobre os seguintes paradigmas, como é mostrado na tabela 4.2:

- **Estruturado:** Programação que segue uma sequência de comandos ou seja o programador expressa seu desejo por ordens. Numa sequência escrita de código.
- **Orientada a objetos:** Segue o modelo baseado na composição e interação entre diversas unidades chamadas de objetos.
- **Orientada a eventos:** É o paradigma de programação que segue um fluxo de controle padronizado que são guiados por indicações externas, chamadas de eventos.

**Tabela 4.2:** Paradigma de Programação

	Estruturado	Orientado a Objetos	Orientado a Eventos
Alice		X	X
Scratch		X	X
AgentSheets			X
Kodu		X	
Squeak Etoys			X
RoboMind		X	

Greenfoot		X	
Stagecast Creator			X
Lightbot			X
Blockly		X	X
Kodable			X
Code.org		X	X

Em relação aos paradigmas de programação podemos destacar que o uso de uma linguagem de programação real nas ferramentas é quase inexistente, sendo elas divididas entre ferramentas orientadas a objetos ou a eventos ou ambas. Aqui fica evidente que o desenvolvimento da lógica computacional é o principal objetivo das ferramentas estudadas deixando a sintaxe em segundo plano.

**Tabela 4.3:** Estruturas de Programação Suportadas

	Condicional	Opera. Lógicos	Laço	Variáveis	Parâmetros	Funções/ Métodos
Alice	X	X	X	X	X	X
Scratch	X	X	X	X	X	X
Agent Sheets	X	X	—	—	—	X
Kodu	X	X	—	—	—	—
Squeak Etoys	X	X		X		X
Robo Mind	X	X	—	—	—	—
Stage Cast	X	X	—	—	—	X
Lightbot	X	X	X			X
Blockly	X	X	X			X
Kodable	X	X	X			X
Code.org	X	X	X			X

As ferramentas que mais se destacam nessa categoria da estrutura de programação é o Scratch e o Alice, por serem mais voltadas ao desenvolvimento elas oferecem ao usuário mais liberdade para programar ao seu modo, como mostra a

tabela 4.3. As ferramentas disponíveis na web são mais limitadas nesse quesito além de seguirem um modelo pré-definido, quase não existindo espaço para o desenvolvimento.

**Tabela 4.4:** Representação do Código

	Texto	Imagens	Fluxograma	Animação
Alice		X		X
Scratch			X	X
AgentSheets		X		
Kodu		X		X
Squeak Etoys	X			
RoboMind			X	X
Greenfoot	X	X		
Stagecast		X		
Lightbot			X	X
Blockly		X	X	
Kodable			X	X
Code.org			X	X

Na representação do código, como é mostrado na tabela 4.4 as ferramentas tentam tornar o mais fácil possível o modo que o código será passado a quem está usando a aplicação. Nesse quesito as ferramentas que utilizam animações têm uma vantagem para com as que usam texto por exemplo.

**Tabela 4.5:** Construção de Programas

	Código em blocos	Linhas de código	Posição de objetos	Comandos
Alice			X	X
Scratch	X			
AgentSheets			X	
Kodu			X	X
Squeak Etoys			X	
RoboMind				X

Greenfoot		X	X	
Stagecast			X	
Lightbot			X	X
Blockly	X			
Kodable			X	X
Code.org	X			

Tabela 4.6: Método de Ensino

	Demonstração	Instruções a cada Etapa	Tutorial
Alice			X
Scratch			X
AgentSheets	X	X	
Kodu			X
Squeak Etoys	X		X
RoboMind	X		
Greenfoot			X
Stagecast Creator	X	X	
Lightbot		X	
Blockly		X	
Kodable		X	
Code.org		X	

São muitas as linguagens e ambientes de programação que surgiram durante os anos tentando tornar mais acessível a programação a iniciantes, ou até mesmo em um nível mais baixo para crianças. Todos esses softwares tentam introduzir a ideia de lógica ao mesmo tempo que possibilitam a criação de coisas que satisfaçam as necessidades de quem estar usando, embora as abordagens e metodologias sejam diferentes. Algumas se preocupam mais com o ensino dos mecanismos de programação de forma mais simplificada, e outras incentivam o desenvolvimento de aplicações.

Pelo fato das ferramentas serem diferentes e também projetadas para diferentes públicos e finalidade, fica praticamente impossível determinar qual a



melhor. Sendo assim foi proposto um modelo de classificação por idade de quem está usando a ferramenta, e pelo grau de dificuldade de uso da ferramenta. Sendo elas divididas entre básica, para as ferramentas que se focam no conceito de operadores lógicos e condicional. Intermediária para as ferramentas que apresentam conceitos de variáveis e funções, e nível avançado para aquelas que abordam conceitos de parâmetros, funções e métodos.

**Tabela 4.7:** Ferramentas por Idade e Nível

Nível	Idade Aproximada	Ferramentas
<b>Básico</b>	3 – 9 Anos	Lightbot, Kodable, Kodu, Code.org, Stagecast Creator, Blockly
<b>Intermediário</b>	10 – 14 Anos	AgentSheets, Alice, Scratch, RoboMind, Squeak Etoys, Kodu, Stagecast Creator
<b>Avançado</b>	14 – 17 Anos	Greenfoot, AgentSheets, Alice, Scratch, RoboMind

Depois da análise dos dados aqui expostos foi possível identificar que há ferramentas que buscam o ensino da lógica voltado a crianças mais novas que ainda não tiveram nenhum contato e também ferramentas focadas no desenvolvimento de qualquer tipo de aplicação que a ferramenta permitir. Todas as ferramentas de auxílio ao aprendizado analisadas abordavam os conceitos básicos de operadores lógicos e condicional. No entanto foi observado a falta de uma ferramenta que atenda a todos os públicos tanto na questão de desenvolver aplicações, quanto no aprendizado da lógica programacional.

Dos softwares selecionados no estudo, para desenvolvimento se destacaram as ferramentas Alice e Scratch, pois são altamente visuais o que facilitam a interação e usam o conceito de blocos de instrução em vez do tradicional texto que facilita no aprendizado de crianças. Para crianças mais jovens é aconselhado a utilização do Code.org e do Lightbot por serem extremamente simples e intuitivas para crianças com pouca idade.

## 5 CONSIDERAÇÕES FINAIS

Foi apresentado neste trabalho uma forma de avaliação de ferramentas de auxílio ao ensino de programação a crianças, sendo que algumas ferramentas apresentam grande potencial para facilitar esse processo. O uso de ferramentas se mostrou produtivo, pois permite que o usuário tenha um contato próximo ao que irá encontrar na sua vida profissional.

Dos softwares selecionados foi observado que apesar de algum terem coisas em comum, cada um trabalha com conceitos de forma isolada em alguns casos. Por esse motivo foi sugerido uma classificação por idade e nível dos mesmos. Esse modelo além de habilitar uma comparação mais adequada entre os softwares com objetivos semelhantes, pode ser usado para a adoção mais adequada a estudantes de acordo com suas características.

Optou-se pelo uso de uma revisão sistemática, visto que a utilização dela pode ser feita uma avaliação mais ampla sobre o assunto, além de dar mais veracidade aos resultados encontrados. Foi encontrada uma dificuldade na elaboração da revisão sistemática por restrições impostas pelas bibliotecas digitais pelo fato da UERN não possuir um contrato com elas, sendo disponibilizado apenas os resumos/Abstracts das obras.

Outra dificuldade encontrada foi achar trabalhos relacionados a ferramentas disponíveis na web como o Code.org, Lightbot e Kodable por serem ferramentas extremamente novas e com poucas avaliações no meio acadêmico.

Por fim a realização desse trabalho faz surgir algumas possibilidades de trabalhos futuros como dar prosseguimento ao desenvolvimento de uma ferramenta que possam tanto auxiliar no ensino de crianças desde os primeiros passos até a etapa de desenvolvimento na mesma ferramenta. E também como outro trabalho futuro ampliar as fontes de pesquisa utilizadas na revisão sistemática, visto que houveram problemas em algumas das fontes de pesquisa selecionados.



GERALDES, Wendell Bento. **Programar é bom para crianças? Uma Visão Crítica sobre o Ensino de Programação nas Escolas**. 2014. Disponível em: <<http://periodicos.letras.ufmg.br/index.php/texolivree>>. Acesso em: 16 mar. 2016.

JESUS E RAABE. Elieser, Andre L. A... **Avaliação Empírica da Utilização de um Jogo para Auxiliar a Aprendizagem de Programação**. 2014. Disponível em: <<http://www.lbd.dcc.ufmg.br/colecoes/sbie/2010/0016.pdf>>. Acesso em: 20 mar. 2016.

JÚNIOR et al, Garibaldi da Silveira. **Análise da Ferramenta de Programação Visual Blockly como Recurso Educacional no Ensino de Programação**. 2012. Disponível em: <<http://ug.iffarroupilha.edu.br/sabtic/files/4.pdf>>. Acesso em: 25 mar. 2016.

JÚNIOR E BONIATI. Rogério Paulo Marcon, Bruno Batista. **LogicBlocks: Uma Ferramenta para o Ensino de Lógica de Programação**. 2015. Disponível em: <<http://eati.info/eati/2015/assets/anais/Longos/L7.pdf>>. Acesso em: 24 mar. 2016.

KELLEHER E PAUSCH. Caitlin e Randy. **Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers**. May 2003.

KITCHENHAM, Barbara. **Procedures for Performing Systematic Reviews**. 2004.

LAHTINEN, E.; Ala-Mutka, K.; Järvinen, H.-M. (2005). "A Study of the Difficulties of Novice Programmers". In: Proceedings do 10th ITiCSE. 14-18.

LEITE, Maira Freitas. **REVISÃO SISTEMÁTICA Um levantamento sobre técnicas e ferramentas que viabilizem o desenvolvimento de sistemas de informação a partir das regras de negócio**. 2011.

MARANHÃO et al, Sergio Souza. **Um Estudo Exploratório dos Games para Introdução ao Pensamento Computacional**. 2015. Disponível em: <[https://www.researchgate.net/publication/275716387\\_An\\_exploratory\\_study\\_of\\_games\\_for\\_introduction\\_to\\_computational\\_thinking?enrichId=rgreq-b891f29b-7215-4e9a-b392-dcdca2b69496&enrichSource=Y292ZXJQYWdIOzI3NTcxNjM4NztBUzoyNDA2NjM1OTc5NDA3MzdAMTQzNDM5MDA4MTM0OA%3D%3D&el=1\\_x\\_2](https://www.researchgate.net/publication/275716387_An_exploratory_study_of_games_for_introduction_to_computational_thinking?enrichId=rgreq-b891f29b-7215-4e9a-b392-dcdca2b69496&enrichSource=Y292ZXJQYWdIOzI3NTcxNjM4NztBUzoyNDA2NjM1OTc5NDA3MzdAMTQzNDM5MDA4MTM0OA%3D%3D&el=1_x_2)>. Acesso em: 25 mar. 2016.

MARTINS, Amilton Rodrigo de Quadros. **USANDO O SCRATCH PARA POTENCIALIZAR O PENSAMENTO CRIATIVO EM CRIANÇAS DO ENSINO FUNDAMENTAL**. 2012. 114 f. Dissertação (Mestrado) - Curso de Programa de Pós-Graduação em Educação, Universidade de Passo Fundo, Passo Fundo, 2012.

Disponível em: <<http://www.upf.br/ppgedu/images/stories/defesa-dissertacao-amilton-rodrigo-de-quadros-martins.PDF>>. Acesso em: 02 mar. 2016.

MEDEIROS, SILVA E ARANHA. Tainá Jesus, Thiago Reis, Eduardo Henrique da Silva. **Ensino de Programação Utilizando Jogos Digitais: Uma revisão sistemática da literatura.** 2013. Disponível em: <<http://seer.ufrgs.br/index.php/renote/article/view/44363>>. Acesso em: 16 mar. 2016.

RAMIREZ-BENAVIDES, Kryscia; GUERRERO, Luis A.. MODEBOTS: Environment for Programming Robots for Children Between the Ages of 4 and 6. **IEEE Revista Iberoamericana de Tecnologías del Aprendizaje**, [s.l.], v. 10, n. 3, p.152-159, ago. 2015. Institute of Electrical & Electronics Engineers (IEEE). <http://dx.doi.org/10.1109/rita.2015.2452692>. Disponível em: <<http://ieeexplore.ieee.org/>>. Acesso em: 15 mar. 2016.

RIBEIRO, Admilo. **Jogo Sério Colaborativo para o Ensino da Programação a Crianças.** 2012. 91 f. Dissertação (Mestrado) - Curso de Engenharia Informática e Computação, Faculdade de Engenharia da Universidade do Porto, Cidade do Porto, 2012. Disponível em: <<http://ticeduca.ie.ul.pt/atas/pdf/237.pdf>>. Acesso em: 25 fev. 2016.

SANTOS, Luís Manuel Mesquita dos. **Modelação de sistemas experimentais em ToonTalk para Ambientes Educativos: Ciclo da Água.** 2009. 209 f. Dissertação (Mestrado) - Curso de Mestrado em Matemática e Ciências da Natureza, Universidade de Trás-os-Montes e Alto Douro, Vila Real, 2009. Disponível em: <<https://repositorio.utad.pt/handle/10348/380>>. Acesso em: 04 mar. 2016.

SILVA, Ronaldo Machado. **O Uso da Linguagem Logo na Educação Infantil.** 2007. Disponível em: <<http://pt.slideshare.net/anapierina/o-uso-da-linguagem-logo-de-programao>>. Acesso em: 12 mar. 2016.

SILVA et al, Eraylson Galdino. **Análise de Ferramentas para o Ensino de Computação na Educação Básica.** 2014. Disponível em: <[https://www.researchgate.net/publication/266387618\\_Analise\\_de\\_ferramentas\\_para\\_o\\_ensino\\_de\\_Computacao\\_na\\_Educacao\\_Basica?enrichId=rgreq-2fe77dfa-09eb-4dbb-b050-d1ca31116695&enrichSource=Y292ZXJQYWdlOzI2NjM4NzYxODtBUzoxNDgzNDY2MzUyOTY3NjhAMTQxMjM4MDAwMjU2Mw%3D%3D&el=1\\_x\\_2](https://www.researchgate.net/publication/266387618_Analise_de_ferramentas_para_o_ensino_de_Computacao_na_Educacao_Basica?enrichId=rgreq-2fe77dfa-09eb-4dbb-b050-d1ca31116695&enrichSource=Y292ZXJQYWdlOzI2NjM4NzYxODtBUzoxNDgzNDY2MzUyOTY3NjhAMTQxMjM4MDAwMjU2Mw%3D%3D&el=1_x_2)>. Acesso em: 12 mar. 2016.

VALASKI E PARAISO. Joselaine, Emerson Cabrera. **Limitações da Utilização do Alice no Ensino de Programação para Alunos de Graduação.** 2012. Disponível em: <<http://www.br-ie.org/pub/index.php/sbie/article/view/1735>>. Acesso em: 22 mar. 2016.

VALÉRIA E FRANÇA. Márcia, César. **Ferramentas de Auxílio ao Aprendizado de Programação: Um Estudo Comparativo.** 2013. Disponível em: <[https://www.researchgate.net/publication/258023558\\_Ferramentas\\_de\\_Auxilio\\_ao](https://www.researchgate.net/publication/258023558_Ferramentas_de_Auxilio_ao)>

Aprendizado\_de\_Programacao\_Um\_Estudo\_Comparativo?enrichId=rgreq-70c77c62-30f6-4e0f-8ddd-2cd600d230e1&enrichSource=Y292ZXJQYWdlOzI1ODAyMzU1ODtBUzoyNjEyMzU2Mzk4NDQ4NjZAMTQzOTI5NDgzODQzMQ%3D%3D&el=1\_x\_2>. Acesso em: 15 mar. 2016.

ZEM-LOPES, Aparecida M. et al. **Uma Revisão Sistemática das Tecnologias da WEB Semântica em Ambientes Educacionais**. II Congresso Brasileiro de Informática na Educação (CBIE 2013). Workshps (WCBIE 2013). 2013.