UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE TECNOLOGIA

MATHEUS CARVALHO MEIRA

APRENDIZAGEM DE LINGUAGEM DE PROGRAMAÇÃO COM METODOLOGIA PBL EM COMPETIÇÕES CIENTÍFICAS COM ROBOCODE

LIMEIRA

2016

MATHEUS CARVALHO MEIRA

APRENDIZAGEM DE LINGUAGEM DE PROGRAMAÇÃO COM METODOLOGIA PBL EM COMPETIÇÕES CIENTÍFICAS COM ROBOCODE

LEARNING PROGRAMMING LANGUAGE WITH PROBLEM BASED LEARNING METHODOLOGY IN SCIENTIFIC COMPETITIONS WITH ROBOCODE

Dissertação apresentada ao Curso de Mestrado da Faculdade de Tecnologia da Universidade Estadual de Campinas, como requisito para obtenção do título de Mestre em Tecnologia na Área de Concentração de Tecnologia e Inovação.

ORIENTADOR: PROF. DR. MARCOS AUGUSTO FRANCISCO BORGES

LIMEIRA

2016

Agência(s) de fomento e nº(s) de processo(s): Não se aplica.

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Faculdade de Tecnologia
Felipe de Souza Bueno - CRB 8/8577

M478a

Meira, Matheus Carvalho, 1983-
Aprendizagem de linguagem de programação com metodologia PBL em competições científicas com Robocode / Matheus Carvalho Meira. – Limeira, SP : [s.n.], 2016.

Orientador: Marcos Augusto Francisco Borges.
Dissertação (mestrado) – Universidade Estadual de Campinas, Faculdade de Tecnologia.

1. Robôs - Programação. 2. Escolas - Exercícios e jogos. 3. Aprendizagem baseada em problemas. 4. Jogos educativos. 5. Aprendizagem por atividades. I. Borges, Marcos Augusto Francisco,1971-. II. Universidade Estadual de Campinas. Faculdade de Tecnologia. III. Título.

Informações para Biblioteca Digital

Título em outro idioma: Learning programming language with problem based learning methodology in scientific competitions with Robocode
Palavras-chave em inglês:
Robots - Programming
Schools - Exercises and recreations
Problem-based learning
Educational games
Active learning
Área de concentração: Tecnologia e Inovação
Titulação: Mestre em Tecnologia
Banca examinadora:
Marcos Augusto Francisco Borges [Orientador]
Marco Antônio Garcia de Carvalho
Eduardo Oscar Epprecht e Machado de Campos Chaves
Data de defesa: 14-12-2016
Programa de Pós-Graduação: Tecnologia

DISSERTAÇÃO DE MESTRADO EM TECNOLOGIA

ÁREA DE CONCENRAÇÃO: TECNOLOGIA E INOVAÇÃO

APRENDIZAGEM DE LINGUAGEM DE PROGRAMAÇÃO COM METODOLOGIA DE APRENDIZAGEM BASEADA EM PROBLEMAS EM COMPETIÇÕES CIENTÍFICAS COM ROBOCODE

Matheus Carvalho Meira

A Banca Examinadora composta pelos membros abaixo aprovou esta Dissertação:

_____

Prof. Dr. Marcos Augusto Francisco Borges
FT/UNICAMP
Presidente

_____

Prof. Dr. Marco Antônio Garcia de Carvalho
FT/UNICAMP

_____

Prof. Dr. Eduardo Oscar Epprecht e Machado de Campos Chaves
FATIPI

Ata da defesa com as respectivas assinaturas dos membros encontra-se no processo de vida acadêmica do aluno.

Dedico este trabalho aos alunos, professores e aos profissionais da área da educação que se preocupam com a constante inovação com experiências e tecnologias para transformar e evoluir a educação.

AGRADECIMENTOS

A Deus,

A Esposa pelo incentivo e dedicação incondicional. Aos pais, meus exemplos de vida. Ao irmão, pelas trocas de experiências.

Ao Prof. Marcos Augusto Francisco Borges pelo aceite da orientação e por acreditar no trabalho. Pela oportunidade em trabalhar no tema da presente pesquisa. Pelo incentivo, apoio e colaboração na execução do trabalho. Pelas conversas, conselhos e dedicação.

A Faculdade de Tecnologia da Universidade Estadual de Campinas, seu corpo docente, coordenação e secretaria da Pós-graduação na representação da servidora Fátima A. Alves.

A equipe do LIAG, que se fez presente em todos momentos da elaboração da pesquisa.

A equipe do IFSP Campus Capivari por colaborar na pesquisa.

“When people learn to play video games, they are learning a new literacy. ”

James Paul Gee

“Se vi mais longe, foi por estar sobre ombros de gigantes. ”

Isaac Newton

RESUMO

Alunos das gerações atuais, inseridos em um contexto de conectividade no universo digital, podem considerar desmotivadores ambientes de ensino tradicionais para disciplinas de programação. Este trabalho apresenta uma abordagem para ensino-aprendizagem de linguagem de programação, a partir do desenvolvimento de competições científicas baseadas no jogo digital educacional Robocode. O objetivo está associado em avaliar se competições científicas com Robocode estimulam aprendizagem de conceitos de programação de computadores em cursos técnicos e superiores da área de informática.

O presente trabalho baseia-se nos conceitos da metodologia de aprendizado baseado em problemas (PBL – *Problem Based Learning*) associado ao Robocode. Técnicas de pesquisa, como estudo de caso e questionários, foram utilizados com objetivo em identificar as potencialidades que o ambiente de ensino-aprendizagem pode proporcionar nas disputas entre equipes envolvendo conceitos de linguagem de programação Java. O trabalho discute a evolução, entre as competições Robocode do LIAG. Com a participação de professores apoiados no PBL, foram determinadas estruturas para os problemas gerados no jogo durante as competições. Professores com papel junto às equipes competidoras, de forma a prover suporte no sequenciamento e levantamento de hipóteses para resolução dos desafios inerentes à competição Robocode.

O resultado do desenvolvimento de ambientes com métodos diversificados de ensino-aprendizagem, apontam as competições como uma abordagem efetiva quando o objetivo é auxiliar o estímulo de alunos com métodos lúdicos e colaborativos na aprendizagem de conceitos de linguagem de programação. A experiência de organização de competições científicas com envolvimento interinstitucional com base no jogo educacional Robocode foi analisada ao longo de dois anos, na Liga 2015 e Robocode Brasil 2016.

Palavras-chave: Robocode, Competições Científicas, Aprendizagem Baseada em Problemas, Jogos Educacionais, Métodos de Aprendizagem.

ABSTRACT

Students of current generations, inserted in a context of connectivity in the digital universe may consider demotivating the traditional teaching environment for programing disciplines. This study presents an approach to teaching-learning programing language from the development of scientific competitions based on the educational digital game Robocode. The goal is associated with evaluating if scientific competitions with Robocode stimulate the learning of computers programing concepts in technical and superior courses of Computer Science área.

The present work is based on problem-based learning (PBL) methodology concepts associated to Robocode. Research techniques, such as case studies and questionnaires were used to identify the potentialities the teaching-learning environment can provide in disputes among teams involving Java programing language concepts. The paper discuss the evolution, between the LIAG competitions Robocode. With teachers participation of supported in the PBL, structures were determined for problems generated in the game during the competitions. Teachers act together with the competing teams, in order to provide support in the sequencing and hypotheses to solve the challenges inherent to the Robocode competition.

The result of the development of environment of diversified teaching-learning methods points out the competition an effective approach when the goal is to assist the stimulation of students with playful and collaborative methods in learning concepts of language programing. The experience of organizing scientific competition with interinstitutional involvement based on Robocode educational game was analyzed over two years, in the League 2015 and Brazil Robocode 2016.

Key-words: Robocode, Scientific Competitions, Problem Based Learning, Educational Games, Learning Methods.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS E ABREVIATURAS

API	*Application Programming Interface*
AVA	Ambiente Virtual de Aprendizagem
CAAE	Certificado de Apresentação para Apreciação Ética
CEP	Comitê de Ética em Pesquisa
DA	Diagrama de Afinidades
EAD	Ensino a Distância
EPL	*Eclipse Public License*
GBL	*Game Based Learning*
GPL	GNU *General Public License*
GUI	*Graphical User Interface*
IA	Inteligência Artificial
IDE	*Integrated Development Environment*
IEEE	*Institute of Electrical and Electronic Engineers*
IFSP	Instituto Federal de Educação Ciência e Tecnologia de São Paulo
IOI	*International Olympiad in Informatics*
JAR	*Java ARchive*
LBD	*Learning By Design*
LDB	Lei de Diretrizes e Bases da Educação Nacional
LIAG	Laboratório de Informática, Aprendizagem e Gestão
MEC	Ministério da Educação
MGBL	*Mobile Game Based Learning*
OBI	Olimpíada Brasileira de Informática
PBL	*Problem Based Learning*
POO	Programação Orientada à Objetos
RPG	*Role-Playing Game*
SGs	*Serious Games*
TI	Tecnologia da Informação
TIC	Tecnologia da Informação e Comunicação
TLCE	Termo de Consentimento Livre e Esclarecido
UDP	*User Datagram Protocol*

SUMÁRIO

1. INTRODUÇÃO

Atualmente, instituições de ensino desenvolvem ações com jogos educacionais digitais nos processos de ensino-aprendizagem para incentivar o aprendizado de programação (Barnes, Powell, Chaffin, Goldwin, & Richter, 2007; Chaffin, Doran, Hicks, & Barnes, 2009). Conteúdos de programação em si não caracterizam apenas assuntos exclusivos das disciplinas específicas de linguagem de programação. Esses assuntos tornaram-se tópicos essenciais para melhor entendimento de aplicativos de uso geral, ou mesmo como ferramentas de auxílio no ensino de variadas abordagens curriculares. A programação ganha o interesse dos jovens, envoltos em tecnologias cada vez mais presentes em suas vidas. A programação consegue despertar a criatividade das pessoas que a utilizam (Celso, Correia, Shimabukuro, & Simonsen, 2008; Chaves, 1998; Valente, 1999). Alunos participantes de disciplinas de programação ou admiradores de tecnologias, muitas vezes têm o anseio de entender a lógica utilizada nos aplicativos atuais. Este trabalho busca apresentar e discutir abordagens interessantes para incentivar a aprendizagem de programação entre alunos de cursos técnicos e superiores da área de informática.

"O método tradicional de ensino é centrado no professor, que possui o papel de sujeito ativo e o aluno passivo no processo de aprendizagem" (Mezzari, 2011). A ênfase do ensino tradicional está na transmissão dos conhecimentos pelo professor, que domina e organiza os conteúdos a serem repassados aos alunos. Com papel de ouvintes, os alunos têm como principal função memorizar informações transmitidas pelo professor (Saviani, 2012). Mesmo em disciplinas de linguagem de programação, no instante em que o professor adota a abordagem do computador como meio para transmitir informação ao aluno, "a máquina está sendo usada para informatizar processos de ensino existentes. Isso facilita a implantação e o uso do computador nas instituições de ensino, porém não quebra a dinâmica do método tradicional" (Valente, 1999).

O Aprendizado Baseado em Problemas (*Problem Based Learning* - PBL) é uma forma de aprender com a utilização de cenários que envolvem problemas da vida real. "É um método que desafia os alunos a aprender e trabalhar em grupos na busca de soluções para os problemas reais" (Hou, 2014). "PBL consiste em uma metodologia pedagógica com base no

construtivismo, na qual a construção do conhecimento ocorre durante a interação entre o ser e o ambiente, com sucessivas acomodações e assimilações" (Rhem, 1998).

O presente trabalho estuda a aplicação de métodos de ensino de conceitos diferenciados aos tradicionais com a determinação de problemas, a partir da disciplina de linguagem de programação, em um ambiente favorável à aplicação da metodologia PBL. A disciplina de linguagem de programação visa uma habilidade fundamental de requisito obrigatório oferecida nos cursos técnicos e superiores na área de informática. Deste modo, os problemas reais devem retratar o ambiente de instituições em que os alunos irão encontrar ao concluir os objetivos curso.

Estudos empíricos demonstram que os alunos podem alcançar ganhos significativos de aprendizagem ao interagir com jogos educacionais em ambientes controlados (Adams, Mayer, MacNamara, Koenig, & Wainess, 2012; Johnson, 2010). "Jogos vão muito além do entretenimento, funcionam como eficientes ferramentas de aprendizagem" (Gallant & Mahmoud, 2008). "Os jogos educacionais digitais têm sido frequentemente sugeridos como promissoras ferramentas pedagógicas" (Prensky, 2012; Tobias, Fletcher, Dai, & Wind, 2011).

Competições científicas para ensino-aprendizagem de linguagem de programação necessitam de estudos prévios para pautar as regras a serem seguidas. "Os jogos de regras pressupõem uma situação problema, uma competição por sua resolução e uma premiação advinda desta solução. As regras orientam as ações dos competidores, estabelecem seus limites de ação, dispõem sobre as penalidades e recompensas. As regras são as leis do jogo" (Maluf, 2008).

O presente trabalho busca associar a metodologia PBL com jogos educacionais em competições científicas[1] para propor e analisar um ambiente de aprendizagem de linguagem de programação. O software, ambiente do jogo educacional Robocode, compõe o cenário de aplicação da metodologia PBL. Robocode consiste em um jogo de programação, composto por um ambiente de desenvolvimento, que permite ensinar linguagem de programação (Java e C#) de modo lúdico (Bonakdarian & White, 2004).

[1] Competições Científicas e Olimpíadas Científicas recebem definições similares de acordo com ICMC da USP e CNPq: "momentos privilegiados para a divulgação científica e para a descoberta e incentivo de novos talentos. O caráter competitivo estimula a inventividade dos alunos e professores"; "buscam incentivar o trabalho em equipe, reforçando hábitos de estudo, o despertar de vocações científicas e os vínculos de cooperação entre equipes de estudantes e professores." (CNPq, 2016; ICMC-USP, 2012).

1.1. Problema da pesquisa

A combinação da metodologia PBL com jogo educacional Robocode em competições científicas pode auxiliar na motivação dos alunos durante o processo de aprendizagem em disciplinas que envolvem linguagem de programação?

1.2. Hipótese da pesquisa

Despertar o interesse no aprendizado com base na linguagem de programação Java, a partir de elementos que facilitam o uso de determinadas estruturas lógicas dispostas no jogo educacional Robocode. Organizar um ambiente propício a competição científica de programação que trabalhe na geração desafios e que despertem o desenvolvimento de pesquisas e trabalhos colaborativos. Estruturar as ações pertinentes ao jogo e ambiente de competição na metodologia PBL para encorajar não somente um conhecimento mais profundo de uma determinada linguagem de programação, mas também de estruturas lógicas que de fato resolvem um problema. A hipótese da pesquisa busca associar jogo educacional Robocode na competição pautada com base na metodologia do PBL para afetar positivamente a motivação dos alunos no aprendizado de disciplinas que envolvam linguagem de programação.

1.3. Objetivos

O objetivo principal deste trabalho é avaliar se competições científicas com base nos conceitos de linguagem de programação Java presentes no Robocode e estruturadas no PBL tem potencial para estimular a aprendizagem em disciplinas que envolvam linguagem de programação.

São objetivos específicos: (1) Propor problemas a partir da ementa da disciplina de linguagem de programação associados ao Robocode; (2) Desenvolver e estudar a competição de programação Robocode; (3) Trabalhar com equipes dos cursos técnicos e superiores da área de informática em instituições públicas e privadas do Brasil; (4) Aplicar o PBL na competição de programação Robocode Brasil 2016 com conceitos presentes nos programas de disciplinas de linguagem de programação; (5) Auxiliar que alunos construam o conhecimento, de acordo

com programa da disciplina, com base no desenvolvimento de soluções para desafios da competição; (6) Avaliar se a metodologia de aprendizagem PBL aliada à competições Robocode motiva os alunos em disciplinas de linguagem de programação.

1.4. Motivação e Justificativa

A crescente retenção dos alunos do primeiro ano dos cursos técnicos, integrados e superiores na área de informática retrata uma preocupação para os campi do Instituto Federal de Educação Ciência e Tecnologia de São Paulo (IFSP). Estudos apontam, para o IFSP, a falta de ações pedagógicas em disciplinas relacionadas a programação de computadores, as que apresentam altas taxas de retenção. A retenção por conteúdo ocorre quando alunos apresentam dificuldades de assimilação e pode provocar a evasão escolar (FEPESP, 2011).

Este trabalho tem como ponto de partida desenvolver e estudar a competição científica Robocode com objetivo de identificar as potencialidades do ambiente de ensino-aprendizagem proporcionadas nas disputas de linguagem de programação. Com base nas observações do ambiente de competição, o trabalho busca avaliar se competições como essas permitem que os alunos desenvolvam estratégias e a programação do jogo com a apresentação das propostas e associação de conceitos a partir dos problemas gerados durante as disputas.

Este trabalho complementa a metodologia aplicada durante as aulas tradicionais. Apresenta as competições científicas com jogo digital para apoiar aprendizado dos alunos em disciplinas de linguagem de programação, para que ele não desista, recupere o conteúdo e principalmente gerar motivação.

Não se pode fixar apenas em meios tradicionais de ensino. É importante combinar diferentes métodos para atrair e incentivar os alunos a conquistar os objetivos propostos no currículo da disciplina. As instituições de ensino podem obter benefícios com os avanços tecnológicos dos *softwares* educacionais para auxiliar o processo de ensino-aprendizagem. Para isso, podem utilizar *softwares* educacionais, do tipo jogos, combinados com métodos para desenvolver ambientes de aprendizagem em disciplinas de linguagem de programação.

1.5. Estrutura do Trabalho

O presente trabalho está organizado na sequência: o Capítulo 2 apresenta o Referencial Teórico com foco nos conceitos relacionados à aprendizagem baseadas em problemas, jogos educacionais digitais e competições científicas; no Capítulo 3, os trabalhos correlatos, ambientes de apoio à aprendizagem de programação, ambientes de competições científicas e o ambiente do trabalho, Robocode são apresentados; o Capítulo 4 apresenta materiais e métodos do trabalho; o Capítulo 5 apresenta históricos de competições Robocode e caracteriza o ambiente de aplicação da competição científica, denominado Robocode Brasil 2016, descrevendo as atividades desenvolvidas e estruturação dos torneios e liga; o Capitulo 6 apresenta resultados e discussões com análise das experiências dos gestores e alunos a partir das técnicas de pesquisa; o Capítulo 7 apresenta a conclusão.

2. REFERENCIAL TEÓRICO

As necessidades de educação estão mudando. É necessário entender que a tecnologia mudou e que os alunos acostumados com um fluxo contínuo de interatividade em diversas aplicações disponíveis nos videogames, *smartphones*, *tablets* etc., em vez de livros e filmes educativos, podem não conseguir ficar sentados para aprender de acordo com modelo tradicional de ensino (Prensky, 2012).

> A cultura digital é a cultura do século XXI. É a nova compreensão de praticamente tudo. O fantástico da cultura digital é que a tecnologia trouxe à tona mudanças concretas, reais e muito práticas em relação a tudo que está acontecendo no mundo, mas também reflexões conceituais muito amplas sobre o que é a civilização e o que nós estamos fazendo aqui (Prado, 2009).

O ambiente do presente trabalho é caracterizado por alunos dos cursos técnicos e superiores da área de informática inseridos em uma cultura digital. Instituições de ensino necessitam estar em constante atualização no que se refere à inovação e a tecnologia do processo de ensino-aprendizagem para manter os alunos motivados.

"Aprendizagem interativa possibilita intercambiar saberes e conhecimento, rompendo com relação unilateral, normalmente estabelecida em ambiente escolar, promovendo um espaço de interação que amplia o número de participantes e as trocas sociais que favorecem o processo de ensino-aprendizagem" (Trescastro & Maria, 2009).

Para diversificação dos métodos de ensino, Balzan argumenta: "Por que não privilegiar discussões em torno de temáticas levantadas junto aos alunos? Por que não prestigiar a aquisição de mentes criativas e inquiridoras, através de debates, de resoluções de problemas extraídos da própria realidade sociocultural?" (Balzan, 1999). Gee também questiona: "Como é possível tornar a aprendizagem, dentro e fora das escolas, mais parecidas com os jogos no sentido que ela use os tipos de princípios de aprendizagem que os jovens veem nos videogames?" (Gee, 2008).

Este capítulo introduz conceitos do ambiente de aprendizagem com jogos educacionais digitais e competições científicas como recursos didáticos para aprendizagem interativa. As seções a seguir têm por objetivo apresentar conceitos importantes para entendimento dos objetivos do presente trabalho. Será apresentado o conceito de aprendizado por *design*

(*Learning by Design* - LBD). Os conceitos que envolvem os jogos educacionais digitais com a caracterização da aprendizagem baseada em jogos serão apresentados. A importância dos jogos na aprendizagem e apresentação do método de aprendizagem baseada em problemas (*Problem Based Learning* - PBL).

2.1. Ambiente de Ensino-Aprendizagem

PBL reúne cenários coletivos, experiências, estratégias etc. na resolução de problemas. Esta seção exibe conceitos dos paradigmas instrucionista, construtivista, construcionista e aprendizagem colaborativa. Ao longo dos paradigmas será possível observar que a busca de conhecimentos prévios ou definições de estratégias em jogos digitais podem se relacionar com construtivismo enquanto discussões grupais de situações que visam cumprir um certo objetivo têm sua relação com aprendizagem colaborativa (Borges, 2004).

O método tradicional de ensino normalmente é denominado de instrucionismo, "um professor/instrutor tem como objetivo transmitir conhecimentos para os aprendizes e, para isso, utiliza uma série de abordagens e recursos didáticos. Como ações, ele deve preparar suas aulas, transmitir os conteúdos e avaliar o grau com que estes conteúdos foram absorvidos pelos aprendizes" (Burd, 1999). "O paradigma instrucionista se baseia em um conhecimento pronto, hierarquizado e compartimentalizado, é transferido do professor ao aluno, que funciona como um repositório de informações" (Skinner, 1993, apud Borges, 2004, p. 39).

Construtivismo tem sua referência em Jean Piaget. Em oposição ao paradigma instrucionista (método tradicional), no construtivismo "o aluno é o sujeito ativo no processo de aprendizagem, o professor age como agente facilitador no processo e orienta o aluno a buscar seus próprios conhecimentos" (Jiménez, 2009). "O facilitador de aprendizado deve ser desempenhado por alguém que conheça os conceitos, porém não é esperado que ele os ensine diretamente. Os aprendizes devem ter liberdade total para explorar o sistema e buscar soluções, sem o controle do facilitador" (Fisher , 1993, apud Borges, 2004, p. 40).

Para Piaget (1978, apud BORGES, 2004, p. 39), a criança desenvolve sua capacidade intelectual interagindo com objetos do ambiente, sem ensino explícito, baseada em uma exploração ativa, onde constrói o conhecimento a partir de um conjunto de problemas motivadores e realistas. Para o construtivismo, o conhecimento é uma função de como um

indivíduo cria os significados a partir de suas experiências e não uma função do que alguém disse que é verdade (Papert, 1986).

Ao optar pelo método construtivista, é possível observar uma vantagem na existência de meios como livros, revistas, televisão, internet etc. disponíveis para consultas. O facilitador (professor) não é o único com acesso aos conteúdos; o aluno possui os mesmos meios para realização de pesquisas e pode se tornar ativo no processo de ensino-aprendizagem (Jiménez, 2009).

Um conceito de modelo mental consiste na forma de enxergar o mundo. É possível observar características do conceito de um modelo mental no instante em que se considera aspectos tecnólogos presentes nos jogos digitais que têm elaboração de estratégias, como o Robocode. A Tabela 1 apresenta um resumo de modelo mental na evolução de uma visão tradicional para uma visão construtivista (Blikstein, 2006).

Tabela 1 - Visão Tradicional Versus Visão Construtivista (Blikstein, 2006).

Visão Tradicional	**Visão Construtivista**
O conhecimento pode ser separado do ato de conhecer.	A existência de conhecimento somente ocorre em seres humanos que constroem sua própria realidade.
O aprendiz adquire conhecimento objetivamente por meio dos sentidos.	O conhecimento é construído subjetivamente pelas pessoas, baseado em experiências anteriores, na forma pela qual refletem e as organizam metacognitivamente. Entende-se metacognitivamente como: consciencializar, analisar e avaliar como se conhece.
O aprendizado envolve a aquisição da verdade que pode ser mensurada por meio de testes.	Se o aprendiz adquire as estratégias que vão ao encontro do objetivo então o aprendizado ocorreu, a medida do aprendizado ocorre por meio de estimativas oriundas de observação e diálogo.

"Dos sistemas baseados no paradigma instrucionista, nos quais pouca ou nenhuma iniciativa e controle são reservados ao estudante, um novo paradigma educacional começou a nortear o desenvolvimento de sistemas computacionais para uso em Educação, fundamentado nas ideias construcionistas de Papert" (Valente, 1999).

"O construcionismo é uma teoria pedagógica com base em reflexões no construtivismo de Piaget, formulada por um de seus discípulos, Seymour Papert" (Cotta, 2002). O construcionismo, "uma derivação do construtivismo, é uma importante referência teórica na utilização das tecnologias digitais na educação" (Ferreira & Duarte, 2012). O construcionismo segue a mesma teoria pedagógica do construtivismo, "*aprender* a *aprender*". Papert denominou

de construcionista a abordagem pela qual o aprendiz constrói, por intermédio do computador, o seu próprio conhecimento (Papert, 1986).

O computador, como instrumento aplicado na educação, pode admitir funções distintas em relação aos paradigmas instrucionista e construcionista:

> Instrucionismo pode ser visto como a informatização dos métodos de ensino tradicionais e o computador têm a função de entregar a informação: alguém implementa no computador uma série de informações que devem ser passadas ao aluno na forma de um tutorial, exercício e prática ou jogo. Já, no construcionismo, o computador requer certas ações que são bastante efetivas no processo de construção do conhecimento. Para ensinar o computador, o aluno deve utilizar conteúdos e estratégias. No caso da resolução de um problema via computador, o aluno tem que combinar estes conteúdos e estratégias em um programa que resolve este problema (Valente, 1999).

A Figura 1 representa o instrucionismo com a interação do aluno no computador em *software* do tipo tutorial. "Um tutorial é um *software* no qual a informação é organizada de acordo com uma sequência pedagógica particular e apresentada ao estudante, seguindo essa sequência ou então o aprendiz pode escolher a informação que desejar" (Valente, 1999). No contexto instrucionista, o aluno está limitado a um prévio conjunto de informações no qual o computador tem o papel de máquina de ensinar. A interação do aluno ocorre com a leitura ou escuta de informações concedidas pelo computador. Em geral, para *softwares* tutoriais, as informações fornecidas são restritas no avanço do material, na escolha da informação e/ou na resposta de perguntas que são digitadas (Valente, 1999).

A Figura 2 representa o construcionismo com a interação do aluno no computador em *software* de programação (ex. programar no robôs virtuais no ambiente Robocode).

> Quando o aprendiz programa o computador (interação com software de programação), este pode ser visto como uma ferramenta para resolver problemas. O programa produzido utiliza conceitos, estratégias e um estilo de resolução de problemas. Nesse sentido, a realização de um programa exige que o aprendiz processe informação, transforme-a em conhecimento que, de certa maneira, é explicitado no programa (Valente, 1999).

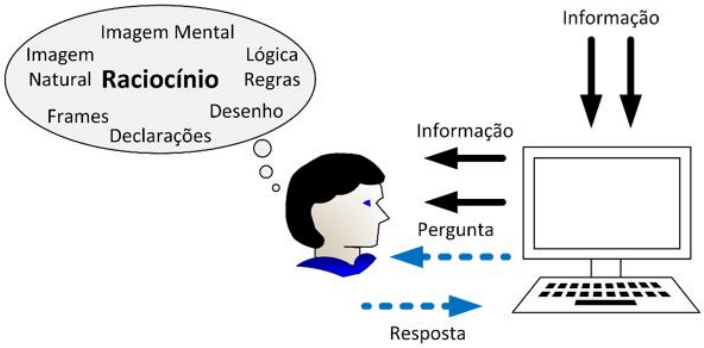


Figura 1 - Interação no Computador com Intrucionismo, adaptado de Valente (1999).

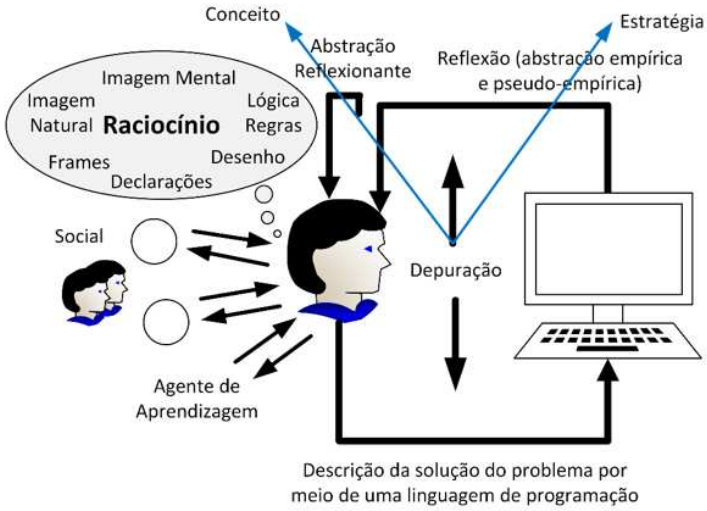


Figura 2 - Interação no Computador com Construcionismo, adaptado de Valente (1999).

O paradigma de aprendizagem colaborativa consiste no processo de ensino-aprendizagem a partir do envolvimento de grupos de trabalhos com objetivo de produzir, com a socialização, conhecimentos, mediados por indivíduos mais experientes (ex. professores, facilitadores) (Soeira & Schneider, 2013). Essa metodologia visa a colaboração mútua no grupo de trabalho, incluindo professor, dentro do processo de aprendizagem, com a proposta em adquirir conhecimento acerca de um dado objeto (Santoro, Borges, & Santos, 1999). “Em um contexto escolar, a aprendizagem colaborativa seria duas ou mais pessoas trabalhando em grupos com objetivos compartilhados, auxiliando-se mutuamente na construção de conhecimento” (Torres & Irala, 2007).

A Tabela 2 exibe um quadro comparativo entre as características presentes nos paradigmas do processo didático do ensino tradicional e da aprendizagem colaborativa (Vanbuel, 1998).

Tabela 2 - Ensino Tradicional e Aprendizagem Colaborativa (Vanbuel, 1998).

Processo Didático do Ensino Tradicional	**Aprendizagem Colaborativa**
Professor responsável pela aprendizagem.	Aluno responsável pela aprendizagem.
Ensino é instrucional.	Ensino aprendizagem é um processo construcional.
Alunos passivos.	Alunos ativos.
Professor instrui e ministra aulas expositivas.	Professor facilita e aconselha (atua como tutor).
Aluno trabalha com material escrito, gravado ou televisionado.	Aluno tem a possibilidade de ter acesso a diversas fontes de informações por meio de tecnologias (livros e revistas *on-line*, televisão, intranet, internet, jogos, etc.).
Aluno recebe informação.	Aluno é o indivíduo que resolve problemas e usa a informação.
Projetos e conquistas individuais.	Trabalho colaborativo.

"A essência da aprendizagem colaborativa está na interação, na qual o arcabouço para aprendizagem não está relacionado com memorização e/ou repetição de conteúdo, mas sim na construção de conhecimentos, conferindo significância ao que é aprendido" (Clementino, 2008).

A Figura 3 apresenta um mapa conceitual, que são "diagramas hierárquicos destacando conceitos de um certo campo conceitual e relações (proposições) entre eles" (Novak & Gowin, 1996). Esse mapa conceitual representa uma visão macro da aprendizagem colaborativa. "A estrutura cognitiva da Figura 3 representa o conteúdo total e organizado de ideias de um dado indivíduo" (Piaget, 1978); ou, "no contexto da aprendizagem de certos assuntos, refere-se ao conteúdo e a organização de suas ideias naquela área particular de conhecimento" (Ausubel, Novak, & Hanesian, 1978). A teoria cognitiva de Piaget defende que a construção de cada ser humano é um processo que acontece ao longo do desenvolvimento da criança.

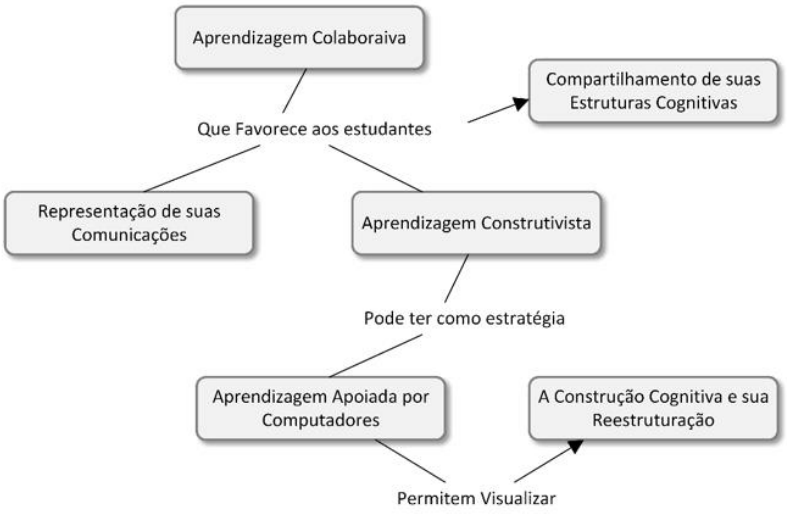


Figura 3 - Mapa Conceitual Macro Aprendizagem Colaborativa (Kwon & Cifuentes, 2009).

O conceito da aprendizagem significativa de Ausubel (1978), “caracteriza-se pela interação entre o novo conhecimento e o conhecimento prévio” (Ausubel et al., 1978). “A aprendizagem significativa requer um esforço do aluno em conectar de maneira não arbitrária e não literal o novo conhecimento com sua estrutura cognitiva existente” (Ausubel et al., 1978). Aprendizagem significativa “é uma teoria cognitiva que busca explicar os meios internos que ocorrem na mente humana em relação ao aprendizado e estruturação do conhecimento” (Ausubel, 2000). Essa aprendizagem visa romper o estigma de que o aluno é uma máquina pronta para receber informações de seu professor. “Ela exibe a necessidade da percepção por parte dos educadores de ferramentas que acelerem o processo criativo, de forma a facilitar a capacidade de aprender e a diversificar a aprendizagem” (Ausubel, 1982).

“O fator isolado mais importante que influencia a aprendizagem é aquilo que o aprendiz já conhece. Descubra o que ele sabe e baseie nisso os seus ensinamentos” (Ausubel et al., 1978). Para haver aprendizagem significativa são necessárias duas condições: (1) Aluno necessita ter disposição para aprender; (2) Conteúdo escolar a ser aprendido deve ser potencialmente significativo. Aprender significativamente consiste em ampliar ideias já existentes na estrutura mental e com isso ser capaz de relacionar novos conteúdos. Quando a aprendizagem é significativa, o aluno consegue avaliar, interpretar e colocar em prática seus conhecimentos. Neste estágio, o aluno tem capacidade de abordar o mesmo tema em situações diferentes (Ausubel, 1982).

2.2. Jogos Digitais Educacionais e Aprendizagem

“Jogo digital é caracterizado por um conjunto de atividades com envolvimento de um ou mais jogadores. Apresenta metas, desafios e consequências. Possui regras e pode envolver aspectos de competição” (Kasvi, 2000). “Jogo eletrônico é uma atividade lúdica formada por ações e decisões que resultam numa condição final. Tais ações e decisões são limitadas a um conjunto de regras e por um universo, que no contexto dos jogos digitais, são regidos por um programa de computador” (Schuytema, 2008). Durante o presente trabalho, é adotado a prática de equivalência de significados para os termos: jogos digitais, jogos eletrônicos, *games* e videogames.

"Os jogos digitais possibilitam o acesso a experiências novas e permitem que se construam modelos da realidade, ou seja, modelos simulados que tornam mais fácil desenvolver coisas no mundo real" (Shaffer, 2006). "Jogos computacionais envolvem conceitos e estratégias que a escola, com todas as suas atividades, não consegue criar. Isso exige do aluno um esforço intelectual e um nível de aprendizagem muito superior às velhas lições de casa" (Papert, 1996).

Os jogos digitais desenvolvem habilidades cognitivas nos alunos porque são elaborados em princípios estruturados de "jogabilidade[2]" (experiências do jogador durante a interação - *gameplay*) e de aprendizagem. Jogos digitais devem conter ambientes interativos com a possibilidade de atrair a atenção dos jogadores proporcionando diferentes níveis de desafios. Contudo, para que os jogos digitais possam ser incorporados como instrumentos educacionais, devem conter características ligadas a aprendizagem (Gee, 2004, 2007, 2008, 2013; Prensky, 2001; Shaffer, 2006). Devem conter objetivos pedagógicos voltados no despertar da criatividade, no desenvolvimento da concentração e raciocínio para proporcionar o ensino-aprendizado prazeroso de um determinado assunto (Gros, 2003).

"Os jogos educacionais digitais têm sido frequentemente sugeridos como promissoras ferramentas pedagógicas" (Prensky, 2012; Tobias et al., 2011). Em virtude das vantagens, os jogos digitais tornam-se cada vez mais presentes como ferramentas de Tecnologias da Informação e Comunicação (TIC) em ambientes educacionais (Gee, 2013). "As características básicas de um jogo educativo incluem orientação, diretrizes para participação, regras do jogo, contexto do jogo ou cenário, e alguma conclusão ou clímax" (Stahl, 1990). "O jogo também pode permitir que os jogadores construam seus próprios objetivos, mas apenas dentro do espaço e regras do qual o jogo foi projetado" (Gee, 2008).

"Um benefício comumente mencionado em relação aos jogos educacionais é seu aspecto motivacional" (Prensky, 2012). Além do aspecto motivador podem ser citados: (1) de acordo com Prensky (2012), benefícios à aprendizagem pelo fato de fornecer um *feedback* imediato; (2) "devido a um ambiente livre de riscos e com *feedback* imediato proporcionado pelo computador, os jogadores são estimulados a explorar, experimentar, cometer erros e tentar novamente" (Gee, 2013).

[2] Jogabilidade: experiências do jogador durante a interação com jogos digitais – *gameplay*. Jogabilidade pode ser caracterizada como a virtude que um jogo possui para ser fácil e intuitivo de se jogar, e está relacionada à curva de aprendizagem de um jogo (Vannucchi & Prado, 2009).

> "O aprendizado baseado em jogos educativos está se tornando uma tendência importante na área de pesquisa em Informática na Educação, pois trata de vários problemas típicos do Ensino, como altas taxas de abandono devido à frustração, falta de motivação para continuar a estudar e a sobrecarga cognitiva do aluno" (Yessad, Labat, & Kermorvant, 2010). "Bons jogos digitais incorporam bons princípios de aprendizagem" (Gee, 2004).

Processos de aprendizagem praticados com o apoio de jogos digitais são designados por aprendizagem baseada em jogos: utilização de jogos digitais em contextos educacionais com objetivo de melhorar e acelerar os processos de aprendizagem, motivando os alunos (Gee, 2007; Prensky, 2012). "GBL (*Game Based Learning* - GBL) consiste em uma abordagem de aprendizagem derivada da utilização de jogos digitais que apresentem valor educacional ou aplicações de *software* que usam games para educação" (Tang, Hanneghan, & El Rhalibi, 2009). Prensky (2012) acredita que a aprendizagem baseada em jogos será considerada uma forma de aprender muito natural. A conclusão do autor tem bases em acontecimentos como a evolução dos automóveis; aviões ou mesmo os computadores. Henry Ford e Orville Wright chegaram a ver evolução dos veículos que se movem a mais de 100 (cem) quilômetros por hora ou jatos supersônicos. Projetistas do ENIAC[3], um dos primeiros supercomputadores do mundo, puderam observar a evolução dos computadores pessoais, *notebooks*, *tablets*, *smartphones* com capacidade de processar milhões de instruções por segundo. Nestes fatos é possível elencar três principais motivos para acreditar na evolução da aprendizagem baseada em jogos: "(1) Está de acordo com as necessidades e os estilos de aprendizagem da geração atual e das futuras gerações; (2) Motiva porque é divertida; (3) Versátil, passível de ser adaptada a quase todas disciplinas, informações ou habilidades a serem aprendidas e, quando usada de forma correta, é extremamente eficaz" (Prensky, 2012).

O termo *Serious Games* (SGs) passou a ser utilizado para identificar os jogos com um propósito específico, ou seja, jogos que transcendam a ideia de entretenimento e ofereçam outros tipos de experiências, como aquelas voltadas ao aprendizado e ao treinamento (Blackman, 2005). *Serious Games* são jogos com objetivos de educar e entreter. Mesmo sem uma definição precisa, SGs visam estimular situações práticas do dia a dia (Susi, Johannesson, & Backlund, 2007).

[3] ENIAC (*Electronic Numerical Integrator And Computer*). Em 1942, o físico John Mauchly propôs uma máquina de calcular totalmente eletrônica. O resultado foi ENIAC, construído entre 1943 e 1945, primeiro computador digital eletrônico de grande escala. http://www.computerhistory.org/revolution/birth-of-the-computer/4/78

> *Serious Game*: competição mental, jogada no computador com regras específicas que usam o entretenimento para treinamento governamental ou corporativo, educação, saúde, políticas sociais e objetivos de comunicação estratégica (Zyda, 2005).

As características dos SGs proporcionam a formação de profissionais, produzem situações para tomadas de decisões em situações críticas e a sensibilização de crianças, jovens e adultos na educação de disciplinas específicas como a linguagem de programação. SGs fazem uso da estratégia, característica marcante na indústria de jogos, para tornar as simulações atraentes aos jogadores bem como, ao mesmo tempo, visa estimular a aprendizagem e a construção de conceitos (Susi et al., 2007; Zyda, 2005).

O termo *Serious Games* pode ser enquadrado como um conceito que promove o desenvolvimento de competências, construção de conhecimentos e atitudes em situações reais. Outras relações ao conceito SGs, estão associadas com aplicações interativas com propósitos lúdicos e definições de metas desafiadoras. Em âmbito educacional, durante o desenvolvimento do processo de aprendizagem, SGs são desenvolvidos em contextos reais ou simulados com interatividade em conhecer o impacto das ações do jogador, por meio de respostas a perguntas ou situações desse cenário virtual. A abordagem do SG tem por objetivo envolver os jogadores, mesmo que cometam erros, para auxílio a determinadas competências em equipes que podem ser: sociais; pessoais; liderança e colaboração. Os SGs auxiliam ainda no entendimento de situações complexas e a busca por estratégias (Susi et al., 2007; Zyda, 2005). SGs são amplamente difundidos dentre as forças militares norte americanas: estima-se que mais de cinquenta diferentes SGs são usadas no apoio ao aprendizado de estratégias e táticas militares (Prensky, 2012). Um desses jogos é "*America's Army*[4]", conhecido como "*AA*" ou "*America's Army Game Project*", é um simulador de guerra desenvolvido pelo exército norte-americano com distribuição gratuita. Consiste em uma franquia disponível para computadores e vídeo games como iniciativa do exército para recrutamento. O jogo simula uma ambiente de guerra virtual com diversas missões tipicamente relacionadas ao treinamento de soldados.

[4] *Americas's Army: The Official Game of the U.S. Army*. Disponível em: https://www.americasarmy.com. Jogo oficial do Exército Norte Americano, baseado no conceito *Serious Games,* lançado em 2002 com mais de dois milhões de jogadores inscritos.

2.2.1. Gerações de Aprendizagem e Relações com Jogos

Nativos digitais são pessoas que nasceram em um universo digital, em contato com internet, computadores e jogos digitais. Nativos digitais compreendem e associam as frequentes mudanças e novidades do mundo tecnológico e se adaptam a esta realidade inconstante na mesma velocidade com que ela se transforma. Imigrantes digitais "são aqueles que não nasceram no mundo digital, mas que em alguma época ficaram fascinados e adotaram muitos ou a maioria dos aspectos da nova tecnologia" (Prensky, 2001, 2012). A Tabela 3 exibe uma relação de características de estilos de aprendizagem entre diferentes gerações (Mattar, 2010).

Tabela 3 - Comparativo entre Estilos de Gerações de Aprendizagem (Mattar, 2010).

Milênio Anterior	**Novo Milênio**
Centrar no trabalho com uma única mídia, mais adequada ao estilo e preferências do indivíduo.	Fluência em múltiplas mídias.
Integração individual de fontes de informação explícitas e divergentes.	Aprendizado baseado em experiências de pesquisa, peneira e síntese coletiva, em vez de localização e absorção de informações em alguma fonte individual melhor.
Experiências de aprendizagem que separam ação de experiência em fases distintas.	Aprendizado ativo baseado na experiência (real ou simulada) que incluem oportunidades frequentes para reflexão.
Usa multimídia ramificada, mas altamente hierarquizada.	Expressão por meio de teias não lineares e associativas de representação em vez de histórias lineares.
Enfatiza a seleção de uma variante pré customizada de uma gama de serviços oferecidos	*Codesign* de experiências de aprendizado personalizadas para necessidades e preferências individuas. Entende-se *codesign* como envolver vários pontos de vista no desenvolvimento e concepção

Conforme o estudo de 2012 de perspectivas da *NMC Horizon Project*, a educação com base em jogos proporciona a melhoria do raciocínio lógico com integração de conceitos pertinentes no plano de ensino das disciplinas com o mundo real (NMC, 2012).

Para serem utilizados com finalidades educacionais, os jogos digitais devem atender objetivos de aprendizagem bem definidos, como promover o desenvolvimento de estratégias no sentido de ampliar a capacidade cognitiva (Gros, 2003). Fatores que contribuem para utilização de jogos digitais como ferramentas educacionais podem ser elencados: (1) Desafio; (2) Fantasia (Imaginários, contextos, temas e personagens); (3) Estímulos sensoriais (visuais e auditivos); (4) Curiosidades e aprendizagens envolvidas (Garris & Driskell, 2002).

De acordo com Prensky (2012), os fatores propostos por Garris & Driskell (2002) são determinantes para motivar alunos a jogar e aprender simultaneamente. São considerados para aprendizagem efetiva (que tenha um sentido para o estudante): (1) Oferta de componentes com interatividade; (2) Feedback; (3) Resolução de problemas e os efeitos do contexto; (4) Comportamentos reflexivos nos jogadores. “Os jogos digitais potencializam a aprendizagem, de forma a permitir a elaboração de reflexões críticas que vão se estabelecendo na prática inerente ao jogo” (Prensky, 2012).

Uma parte pertinente da aprendizagem com utilização de jogos digitais se concretiza fora do ambiente do jogo, numa reflexão em relação a experiência, apresentada na Figura 4 com o resumo do raciocínio (Garris & Driskell, 2002).

Figura 4 - Processo de Aprendizagem nos Jogos (Garris & Driskell, 2002).

2.3. Competições Científicas de Programação

Competições científicas de programação oferecem abordagens motivadoras aos alunos para introduzir conceitos de programação e encontrar novos talentos. Motivar alunos a aprender consiste em um importante tema de discussão entre professores. Neste contexto, as competições apresentam-se como abordagens que oferecem ampla gama de possibilidades para auxiliar e envolver alunos no aprendizado de conceitos de programação. Essas possibilidades, são convertidas em contextos de atividades e desafios dentro competições de programação com o propósito de envolver e motivar alunos (Hakulinen, 2011). “Motivação consiste em um processo responsável pela intensidade, direção e persistência dos esforços de um indivíduo para alcance de uma determinada meta” (Robbins, 2011).

Estudos apontam que as competições científicas, quando estruturadas, podem ser consideradas métodos de estímulo para que alunos aprimorarem seus conhecimentos em áreas específicas, como a programação de computadores. Quando motivados a participar das disputas em instituições de ensino, alunos tendem a estar envolvidos com a missão de vencer os objetivos inerentes da competição (Robbins, 2011).

Competições de programação estão cada vez mais populares e não são desenvolvidas somente com a escolha de um jogo, mas como abordagens estruturadas aplicadas em processos de ensino-aprendizagem. Essas abordagens podem ser aplicadas em diferentes cenários: (1) Na área acadêmica, para formação técnica básica de profissionais, como fonte de motivação para desenvolvimento de pesquisas; (2) Em empresas, como forma de treinamento coletivo, desenvolvimento de novas ou específicas habilidades ou como seleção de funcionários.

Há várias competições de programação realizadas em todo mundo e seus propósitos fundamentais podem variar. Os objetivos, por exemplo, podem ser: testar conhecimento dos concorrentes; possibilitar auxílio na aprendizagem; encontrar talentos; ou mesmo promover a computação. Como propósitos das competições variam, atividades envolvidas dentro das competições podem conter diferentes características, dependendo do tema a ser abordado. Essas características são influenciadas pelo público-alvo e prévio conhecimento necessário, dados que podem apresentar diferenças em cada competição (Hakulinen, 2011).

São alguns dos objetivos das competições científicas: aproximar uma disciplina específica ao cotidiano dos alunos; apresentar aplicações práticas na resolução de problemas reais ou simulados; e envolver de modo natural o interesse dos alunos. As competições também despertam novos vínculos dos alunos perante a escola e incentivam trabalhos em equipes com definições de estratégias cooperativas de aprendizagem. Uma competição trabalha valores afetivos como a autoconfiança e autoestima do aluno, na medida da evolução das disputas, associando-as a sua capacidade de interagir no grupo e resolver problemas. Atualmente, as competições também são empregadas de acordo com Science Olympiad[5] (2016) para atrair estudantes do sexo feminino nas carreiras de áreas como ciência e tecnologia (ScienceOlympiad, 2016).

[5] Science Olympiad: https://www.soinc.org, organização sem fins lucrativos dedicada a melhorar a qualidade da educação em ciência.

Em qualquer competição é importante, e muitas vezes necessário, que o aluno passe por uma temporada de preparação para conquistar bons resultados. Nas competições científicas de programação, definições de grupos de estudos são consideradas maneiras de estimular alunos a estudarem conceitos e, consequentemente, de apoiar na preparação para disputas. A competição permite integração entre alunos de diferentes períodos ou mesmo instituições (Fassbinder, Paula, & Araujo, 2012).

Inovações com as tecnologias digitais trouxeram a maratona de programação hacker, conhecida como "*Hackathon*". Essas maratonas são conhecidas como eventos em que programadores de computadores e outros profissionais envolvidos (desenvolvedores, *designers*, educadores e inventores em geral) colaboram durante um curto período de tempo (máximo de 48 horas) em projetos de *software*. *Hackathon* é combinado a partir das palavras da língua inglesa: *hack* como programar de modo excepcional e *marathon* de maratona, no sentido de exploração e investigação da programação. O termo surgiu em 1999, decorrente da união de um pequeno grupo de desenvolvedores para evitar problemas legais de regulamentação de *software* criptografados nos EUA (Estados Unidos da América) junto ao Sistema Operacional (SO) *opensource* OpenBSD[6] (Briscoe & Mulligan, 2014).

Maratonas de programação no estilo *hackathon*, atualmente, têm sido amplamente difundidas como uma das formas diferenciadas de desenvolver projetos considerados inovadores dentro das empresas. São maratonas para desenvolvimento de uma solução que atenda a um fim especifico ou projetos com temáticas livres realizados em um único evento. *Hackatons* geralmente são destinadas à jovens desenvolvedores com tarefas centradas em desenvolver protótipos funcionais em busca oportunidades sociais, profissionais e premiações (Richard, Kafai, Adleberg, & Telhan, 2015). Também são características de maratonas *hackathon* trabalhos motivados por desafios de modo lúdico para resolução de problemas. *Hackathon* vem sendo considerada como um evento de impacto significativo de inovação da cultura digital (Briscoe & Mulligan, 2014).

[6] https://www.openbsd.org

2.4. Aprendizagem por *Design (Learning by Design – LBD)*

Com base no método PBL, foi desenvolvida a abordagem de ensino denominada Aprendizagem por Design *(Learning By Design* - LBD). A LBD é caracterizada por uma abordagem construtivista para promover o aprendizado, podendo apoiar o processo de ensino-aprendizagem para desenvolver habilidades práticas e na compreensão necessária para solução dos problemas (Kolodner, Hmelo-Silver, & Narayanan, 1996).

O LBD aborda necessidades emergentes nos alunos nativos digitais com aprendizagem no desenvolvimento prático (Gee, 2013). Na LBD, os professores estão engajados no desenvolvimento das práticas pedagógicas e durante o processo de aprendizagem podem auxiliar os alunos a comparar ideias e identificar o que eles precisam aprender para prosseguir na resolução do problema (Kolodner et al., 1996).

Jogos digitais que têm a capacidade de desenvolver diversas aprendizagens são chamados de "bons jogos" na visão de Gee (2013). Nos "bons jogos", o jogador constrói ou assume um personagem que se move em um mundo ficcional resolvendo diversos problemas inerentes aos objetivos do jogo. Portanto, o aprendizado pode ser obtido a partir do desenvolvimento de abordagens práticas LBD nos "bons jogos" educacionais.

Os princípios da abordagem LBD são divididos em 3 (três) categorias: (1) Capacitar os Alunos (*Empowered Learners*); (2) Aprendizado Baseado em Bons Problemas (*Good Problem Based Learn*); e (3) Entendimento Profundo (*Deep Understanding*). Para apoiar a adoção do LBD , os jogos educacionais podem ser eficientes, com práticas pedagógicas planejadas, dentro do cenário de aprendizagem (Gee, 2007, 2013). A seguir são indicadas as características das 3 (três) categorias dos princípios do LBD.

A primeira categoria de princípios LBD, *Empowered Learners,* indica que "bons jogos" devem capacitar os alunos a identificar o quanto as decisões tomadas podem afetar as ações do jogo. O planejamento da capacitação deve buscar despertar situações de interesse dos alunos com base no jogo educacional. A capacitação e a motivação dos alunos, dentro do jogo, afeta o modo de abordagem e o desempenho na resolução dos problemas. A Tabela 4 indica princípios envolvidos no *Empowered Learners* com as situações de interesse.

Tabela 4 – Princípios do *Empowered Learners* (Gee, 2013).

Princípios	Características
Agente *(Agent)*	Participante
Produção *(Co-design)*	Os jogadores são produtores, não apenas consumidores. Produzem os jogos com a ações e decisões executadas.
Customização *(Customization)*	Configurar o jogo conforme as experiências, dificuldades, tipos de jogadores, estratégia e categorias. Diferentes maneiras de personalizar os problemas e estratégias ao estilo de aprender a jogar (ex. ao escolher um nível de jogo: *easy* ou *hard*)
Identidade *(Identity)*	Quais são as questões cruciais: onde serão aplicados os conteúdos aprendidos se resolver o problema? Jogos são muito bons em criar identidade, se fizer isso a recompensa é, ou chega em algum lugar, ou ganha bônus, passa de estágio etc. Outra questão é identificar a identidade dos jogadores. Bons jogos digitais cativam os jogadores, fazem com que firmem um compromisso com as metas estabelecidas.
Manipulação *(Manipulation)*	Trabalhar o ambiente no sentido de envolver os alunos em um ambiente de aprendizagem. Explorar as possibilidades do ambiente do jogo em prol da aprendizagem. Definir as principais variáveis que poderão auxiliar no aprendizado.

Para segunda categoria, os princípios estão relacionados com o desenvolvimento de um bom problema, conforme apresentado na Tabela 5. O trabalho relaciona o referencial teórico da presente categoria, *Good Problem Based Learn*, com o PBL para criar as problematizações que envolvem os alunos durante a competição. “Bons jogos digitais apresentam aprendizado baseado em bons problemas” (Gee, 2013).

Deep Understanding caracteriza a terceira categoria dos princípios LBD, implicam em adquirir uma compreensão não apenas superficial de conteúdos propostos durante o jogo (Gee, 2013). A pirâmide de aprendizagem apresentado na Figura 5 exibe características da aprendizagem superficial e aprendizagem profunda. “Aulas expositivas ou leitura são somente capazes de provocar uma aprendizagem superficial, associado a níveis de aprendizado mais baixos” (Dale, 1969). A Tabela 6 indica os princípios da categoria *Deep Understanding* (Gee, 2013).

Em uma visão psicológica da pedagogia aplicada à linguagem de programação, podem existir negligências nas estratégias para formação de novos programadores. Em caso de falhas no processo de aprendizagem, geralmente programadores iniciantes podem apresentar limitações com um conhecimento superficial ou frágil do assunto. Esse conhecimento fragilizado indica que o aluno sabe, mas não consegue usar quando necessário (Winslow, 1996). “Uma abordagem superficial à aprendizagem é inadequada em uma disciplina de assunto prático, defendendo uma abordagem ‘*learn by doing*’” (Jenkins, 1998).

Tabela 5 – Princípios do *Good Problem Based Learn* (Gee, 2013).

Princípios	Características
Resolução de problemas *(Problem Solvem)*	Foco na resolução do problema. Utilização de fatos, informação e instrumentos necessários para resolução. Jogadores veem sentido de propriedades em relação ao que estão trabalhando.
Boa ordenação de Problemas *(Well-ordered Problems)*	Estratégia para boa Ordenação de problemas. Levantar as Probabilidades de resolução. Problemas são ordenados de forma a conduzir os jogadores a formularem hipóteses que funcionem na sua resolução.
Sequenciamento *(Sequencing)*	Estabelecer o nível dos problemas e uma sequência de resolução. Não iniciar o jogo em um nível mais difícil. Desenvolvimento e apresentação de problemas em níveis iniciais envolvem o jogador e podem resultar em sucesso posterior.
Desafio Agradável *(Pleasantly)*	Um desafio que possa trazer realização em resolver, agradável. Deve indicar que com esforço, empenho é possível resolver. Deve envolver os alunos de modo a perder o senso do tempo.
Ciclo da Avaliação *(The Cycle of Expertise)*	Modo como cada jogador se torna um *expert*. Envolvem um conjunto de problemas para atender: (1) Desafio (*challenge*); (2) Prática (p*ractice*); (3) Conhecimento (k*nowledge*); (4) Domínio, Controle (m*astery*). Ao concluir, criar novos níveis (*levels*) de desafios.
Informação na hora certa e sob demanda *Information Just in Time and On Demand*	Com uma pequena parte da informação é possível criar uma base para estruturar o assunto que envolve o problema. Não lidar com grande número de palavras fora do contexto: informações verbais "na hora certa", no momento em que podem utilizadas, "a pedido" quando houver a necessidade. Com informações sob demanda é possível criar pontos de controle (Ex. "deste ponto em diante até etapa 10 resolvido, OK!" Aguardar outras instruções a partir de outras informações sob demanda).
Delimitação do Ambiente *(Fish tank)*	Estabelecer um "aquário" para abordar a complexidade dos problemas. Se o sistema for muito complexo e o aluno tentar aprender tudo de uma única vez pode ficar sobrecarregado. Iniciar com um ambiente equalizado onde consta um número de variáveis que o aluno possa trabalhar sem comprometer o rendimento e/ou perder o interesse. Se existirem muitas variáveis acrescentar de modo gradativo.
Delimitação do Problema *(Sandboxes)*	Local para explorar as possibilidades. Criar um espaço "Caixas de Areia" para delimitar o problema de modo a permitir que os alunos possam explorar esta área com segurança (dentro dos assuntos da proposta do problema).

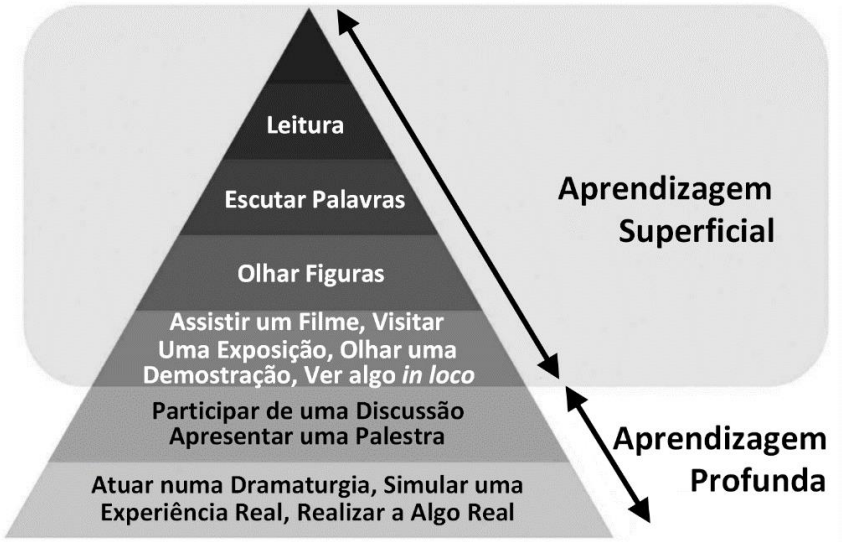


Figura 5 - Pirâmide de Aprendizagem (Dale, 1969).

Tabela 6 – Princípios do *Deep Understanding* (Gee, 2013).

Princípios	Características
Habilidades como estratégia *(Skills as Strategies)*	Para desenvolver as habilidades básicas é necessário a prática. Utilização de habilidades para cumprir as metas.
Pensamento Sistemático *(System Thinking)*	Entender a complexidade do problema. Conjunto de regras que permita entender a origem dos efeitos com base em decisões tomadas (Origem dos fatos baseado nas decisões). Em jogos de múltiplos jogadores é necessário pensar não somente nos aspectos do mundo do jogo, mas nas ações dos outros jogadores.
Meaning as Action	Significado como ação. Jogos digitais tem a característica de contextualizar o significado das palavras em termos das ações, imagens, diálogos etc. Liga os jogadores não apenas em sentidos verbais, mas as experiências a que eles se referem.

2.5. Aprendizagem Baseada em Problema (*Problem Based Learning – PBL*)

PBL é um método com mais de 50 anos, inspirado em exemplos de experiências das escolas de medicina da Universidade de McMaster (Canadá) e Universidade de Maastricht (Holanda), posteriormente difundido e recomendado pela Sociedade das Escolas Médicas em países da África, Ásia e América Latina. Mesmo com sua origem na área da saúde, atualmente PBL é encontrada em diversas disciplinas no ensino superior, médio e básico (Berbel, 1998).

Os processos gerais do PBL envolvem discussões entre os alunos sobre os problemas de forma a definir o que sabem, obter os objetivos de aprendizagem e atribuir tarefas para continuidade de sua resolução. As características que definem o PBL incluem a aprendizagem conduzida por problemas abertos que representem desafios, nos quais os alunos trabalham em grupos colaborativos e professores atuam como "facilitadores" de aprendizagem (Hmelo-Silver & Barrows, 2006).

"PBL fundamenta-se em resultados de pesquisas educacionais, especialmente na área de psicologia cognitiva, que indicam que o trabalho dos alunos com a vida real, particularmente em grupos, favorece a aprendizagem" (Filho & Ribeiro, 2009). A psicologia cognitiva está relacionada ao estudo dos processos mentais que influenciam no comportamento de indivíduos e o desenvolvimento cognitivo. No PBL, o problema deve ser real ou potencialmente real, podendo ser trabalhado em cenários reais ou fictícios projetados pela provedora de ensino. No PBL, alunos, em pequenos grupos (4 a 12), devem analisar problemas com base nos conhecimentos prévios. Os estudantes devem trabalhar de modo a perceber as questões

envolvidas e, de uma forma estruturada, identificar o que é necessário saber e de que maneira vão descobrir. “A adoção do PBL é justificada por seus idealizadores como uma resposta à percepção dos professores de que os alunos estavam saindo do curso com muitos conceitos, mas pouca capacidade de utilizá-los e integrá-los à prática cotidiana” (Barrows, 1996).

Existem cinco características para o desenvolvimento de bons problemas PBL: (1) Devem envolver o interesse dos alunos, motivá-los a sondar a compreensão mais profunda dos conceitos envolvidos; (2) Projetados com múltiplos estágios; (3) Complexos o suficiente para que haja a necessidade de cooperação de todos os membros do grupo para eficaz solução do problema; (4) Devem ser abertos para comportar várias respostas igualmente válidas (mesmo na ausência de uma única resposta correta, pode haver uma melhor solução); (5) Os objetivos do conteúdo da disciplina devem ser incorporados no problema (Duch, 2001).

“A característica mais importante no PBL é o fato de uma situação-problema sempre preceder a apresentação dos conceitos necessários para sua solução” (Filho & Ribeiro, 2009). A Figura 6 apresenta algumas características que respondem aos fundamentos e objetivos educacionais inerentes ao método PBL.

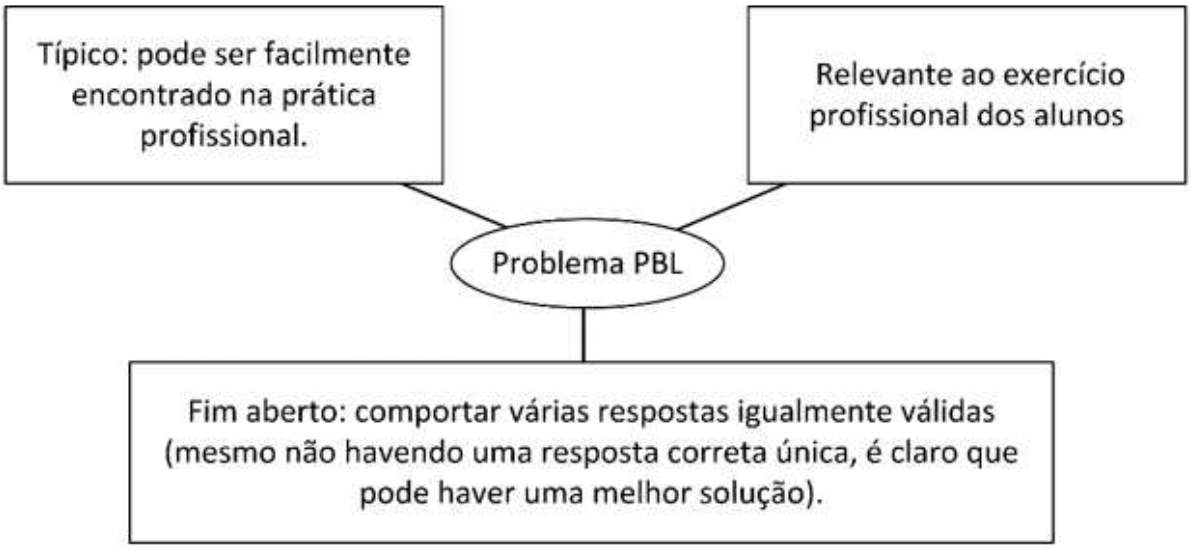


Figura 6 - Fundamentos e Objetivos Educacionais PBL (Filho & Ribeiro, 2009).

O processo de aprendizagem, com base na metodologia PBL, parte da apresentação de problema para grupos de alunos sem uma prévia instrução sobre a sua resolução. O objetivo do problema é instigar os estudantes a pesquisar acerca de determinados conteúdos relacionados a

uma ementa previamente estruturada. Os alunos devem analisar o problema e determinar quais questões se apresentam e que informações são necessárias para solução. Um estudo autônomo, antes da reunião em grupos, a fim de compartilhar as descobertas necessárias na aplicação da resolução, faz parte do processo (Gil, 2008).

Os processos gerais do PBL envolvem: (1) Os estudantes discutirem os problemas; (2) Investigar o que sabem; (3) Gerar hipóteses; (4) Obter objetivos de aprendizagem e (5) Atribuir tarefas individuais para prosseguir na resolução do trabalho. O professor tem a função de facilitador do processo de aprendizagem, por meio de perguntas abertas, deixando de ser o responsável por prover o conhecimento (Hou, 2014). O professor fornece *feedback* sobre o processo de aprendizagem e dinâmicas de grupo. "As características que definem o PBL geralmente incluem a aprendizagem conduzida por problemas abertos e desafiadores, trabalho de alunos em pequenos grupos colaborativos e professores como 'facilitadores' da aprendizagem." (Hou, 2014).

> Um dos fatores determinantes do PBL diz respeito ao grau de estruturação dos problemas, porque um problema estruturado de forma incompleta faz com que os alunos, à medida que investigam uma solução, descubram a complexidade do problema e percebam que ele pode corresponder a várias soluções (Delisle, 2000).

Um problema no PBL, quando estruturado de forma incompleta, exige a pesquisa de mais informações além das fornecidas com objetivo de compreender os problemas na definição das estratégias que serão usadas para sua resolução. Quanto menos estruturado se apresentar um problemas, exigirá mais do aluno em relação de desenvolver habilidade para solução do problemas e, consequentemente, conduzir seu próprio aprendizado (Filho & Ribeiro, 2009).

Em contraste com os métodos tradicionais de ensino, os alunos do PBL assumem maior grau de responsabilidade perante sua aprendizagem. A dinâmica das interações de grupos e estudos independentes auxiliam o estudante a conquistar níveis avançados de compreensão. PBL permite desenvolver habilidades de aprendizagem, formação de conhecimento e promover práticas de construção de equipes (Hou, 2014).

2.6. Considerações Finais

O presente capítulo contextualizou o trabalho, discutindo as características do ensino intrucionista, construcionista e aprendizagem colaborativa. Foram apresentadas referências relacionadas aos jogos educacionais digitais e competições científicas de programação. Foram apresentadas as metodologias de aprendizagem baseada em jogos e aprendizagem baseada em problema, discutindo a importância dos jogos educacionais no aprendizado por desenvolverem habilidades cognitivas nos alunos. Foi introduzida a abordagem do *Learning by Design* no sentido de viabilizar a aprendizagem no desenvolvimento com prática. O próximo capítulo apresenta ambientes educacionais de aprendizagem.

3. AMBIENTES EDUCACIONAIS DE APRENDIZAGEM

Disciplinas como “Introdução à Programação” ou “Lógica de programação” são consideradas tradicionais e fazem parte da ementa obrigatória para muitos alunos que estudam em cursos de computação e áreas afins. Com estabelecimento de competências para TIC (Tecnologia da Informação e Comunicação) nos últimos anos, o conceito de programação migrou para escolas de ensino técnico, médio e até mesmo nível fundamental. Em termos de competências para o século XXI, as disciplinas de programação exigem habilidades necessárias para sobreviver na era da informação em uma sociedade globalizada (Kalelioglu & Gulbahar, 2014).

A utilização de jogos educacionais digitais para ensino de programação de computadores é prática conhecida e adotada por muitos professores e instituições ensino. Além da diversificação de recursos, essa prática com envolvimento de jogos digitais pode despertar o aspecto motivacional para estudo de disciplinas como a linguagem de programação. Com a crescente popularização dos jogos digitas nas salas de aulas, naturalmente surgem competições científicas de programação como extensão da utilização dos recursos presentes nos desafios encontrados nos jogos. Atualmente muito populares, as competições científicas são utilizadas além do objetivo geral do jogo, mas como importantes ferramentas para favorecer o processo de ensino-aprendizagem. Na área acadêmica, competições com jogos digitais em áreas como a linguagem de programação estão cada vez mais consolidando-se como uma abordagem efetiva para desenvolvimento de habilidades da área de computação (Nishimura, Kawasaki, & Tominaga, 2011).

O presente capítulo apresenta trabalhos correlatos, ambientes educacionais para apoio na aprendizagem de conceitos de programação e aprendizagem nas competições científicas. Será contextualizado o jogo educacional digital com *framework* Robocode, ambiente de aplicação das competições do presente trabalho.

3.1. Trabalhos Correlatos

A presente seção apresenta trabalhos correlatos relacionados à difusão da aprendizagem, com foco na programação, entre jovens com base em *Serious Games*. É apresentado um trabalho relacionado ao aprendizado de linguagem de programação com objetivo em treinar equipes para disputas em maratonas de programação. São discutidas inciativas em trabalhar com PBL e o ambiente educacional do Robocode para incentivar ensino-aprendizagem de conceitos de linguagem de programação em cursos superiores.

Code.org na Difusão da Programação

O trabalho desenvolvido pela organização norte americana Code.org[7] segue tendências propostas em um cenário mundial dedicadas a expandir o ensino-aprendizagem de ciência da computação de modo a incentivar o acesso ao conhecimento de programação entre jovens. A organização conta com dezenas de parcerias junto a empresas com representatividade no cenário mundial e com apoio do governo norte-americano. É possível citar ações de participação do presidente norte-americano em um dos programas do Code.org denominado "*Hour of Code*". No Brasil, conhecido como "Hora do Código", Code.org se enquadra nas ações que têm por objetivo permitir e incentivar a compreensão de conceitos de programação de modo lúdico (Code.org, 2016).

***Serious Games* para crianças com Hemofilia**

Jogos educacionais digitais têm desenvolvido papel fundamental quando envolvem diferentes abordagens de aprendizagem. Na área médica, foram utilizados jogos sérios (*Serious Games* - SGs) para várias aplicações como, por exemplo, para informar, esclarecer e incentivar comportamento adequados para crianças com hemofilia. Denominado de Hemotion, o jogo apresenta um modo lúdico e motivador de educar crianças diagnosticadas com hemofilia. O SG Hemotion pode ser apresentado como uma abordagem efetiva quando o objetivo é auxiliar crianças a entender melhor, não somente sobre a natureza da doença, mas também acerca de

[7] A Code.org criada em 2013 como uma organização sem fins lucrativos dedicada a expandir o ensino de ciência da computação, tornando-a disponível em mais escolas e aumentando a participação de mulheres e negros, os quais são sub-representações nessa área. Tem como visão que todos os alunos, de todas as escolas, devem ter a oportunidade de aprender programação de computadores. Acredita que a Ciência da Computação deve fazer parte do currículo básico, ao lado de outros cursos biologia química ou matemática (Code.org, 2016).

como lidar com situações de risco que podem agravar o quadro clínico (Matsunaga, Moraes, Borges, Matta, & Ozelo, 2014).

Treinamentos para Maratonas de Programação

A pesquisa realizada por Piekarski et al. (2015) apresentou a experiência de um projeto de extensão, a partir de conceitos que envolvem maratonas de programação. A pesquisa teve por objetivo o treinamento de alunos em linguagem de programação para disputa de maratona de programação. Essa experiência visou a seleção de equipes para participar da primeira fase de umas das maratonas de programação mais importantes realizadas no Brasil (Maratona de Programação Sociedade Brasileira de Computação – SBC). Foram estabelecidos encontros para oferecer atividades lúdicas que favoreçam o aperfeiçoamento das habilidades em programação de computadores. Nos encontros foram estabelecidos problemas de competições anteriores e disponibilizados em um ambiente de avaliação automática para que as equipes pudessem testar suas soluções. As competições foram simuladas nos mesmos moldes das competições oficiais da SBC. De modo geral, as atividades realizadas tiveram o objetivo geral associado em despertar o interesse dos alunos na maratona de programação da SBC. Os objetivos específicos visaram estimular a resolução de problemas computacionais e trabalho em equipe (Piekarski, Miazaki, Hild, Mulati, & Kikuti, 2015).

Abordagem PBL e Robocode no Ensino-aprendizagem de Programação

Os princípios que regem PBL foram aplicados no ensino-aprendizagem de linguagem de programação em um curso superior, Ciência da Computação, com apoio do ambiente educacional Robocode. Os estudos exibem a necessidade de construir bases que permitam aos estudantes buscar o conhecimento. Buscou-se envolver os alunos e usar o ambiente para despertar o interesse na compreensão dos conceitos introduzidos. A experiência indicou que, para obter o máximo de benefícios a partir de resolução de problemas, um jogo educacional funciona como uma ferramenta no despertar do interesse (O'Kelly & Gibson, 2006).

Utilização de Problemas da Maratona de Programação e Juízes Eletrônicos como Estratégia de Ensino em um Curso de Graduação em Engenharia de Software

A pesquisa retrata as dificuldades encontradas pelos professores no ensino de programação em cursos superiores da área de computação. Atribuem essas dificuldades como fatores que contribuem para o alto índice de evasão dos alunos nos cursos de computação. Do

lado dos alunos, são exibidos diversos motivos para a evasão, dentre a falta de conhecimento acerca dos fundamentos básicos do curso, dificuldades na lógicas para resolução de problemas, falta de dedicação e interesse na linguagem de programação ou mesmo dificuldade de aprendizado com a estratégia de ensino adotada (Pereira, Costa, Sales, & Sales, 2016).

O método da pesquisa centrou na estratégia de utilização dos próprios problemas apresentados nas maratonas de programação bem como os juízes eletrônicos para elaboração de problemas que abordassem o conteúdo da disciplina. Foram realizadas análises de desempenho dos alunos em diferentes disciplinas que envolvem assuntos de programação de computadores. Como resultado, segundo a pesquisa de Pereira et al. (2016), foi possível observar melhoria no desempenho em relação ao métodos tradicional de ensino para o método a partir de elaboração de problemas com base na maratona de programação. O estudo atribui a melhoria do desempenho influenciado pela motivação dos alunos participantes, no instante em que foram apresentados a problemas próximos de sua realizada, passaram a enxergar a programação como uma ferramenta de trabalho com aplicações reais.

3.2. Ambientes Educacionais no Apoio à Aprendizagem de Programação

"Se é difícil desenvolver habilidades de programação em nível de ensino superior, qual deveria ser a estratégia mais adequada para lidar com alunos de ensino primário e secundário?" (Kalelioglu & Gulbahar, 2014). Marques e Marques (2012) sugerem como resposta a utilização do *Scratch* para ensino de linguagem de programação para resolução do problema: "Os alunos desenvolvem todas habilidades referentes à lógica/programação e logo descobrem problemas do qual sentem a necessidade de resolver a fim de avançar no projeto" (Marques & Marques, 2012). Algumas características do *Scratch* e exemplos de jogos educacionais para ensinar programação serão visualizados nos próximos itens do presente tópico.

RoboCup 2D / 3D

O RoboCup trata uma competição em nível mundial que acontece todos os anos com propósito educacional no auxílio da aprendizagem de programação de computadores. Este torneio visa estabelecer desafios e resolução de problemas com envolvimento de tecnologias e

metodologias da IA, Robótica e áreas afins (Robocup, 2016). "A RoboCup reúne a necessidade de lidar com a complexidade do mundo real criando robôs simulados que demonstrem um alto nível de competência para a realização de tarefas, como chutar, interceptar, entre outros" (Kitano & Veloso, 1997).

Atualmente, existe a competição denominada "*Soccer Simulation League"* regida pela Robocup2016.org[8] que abriga duas diferentes Ligas: Figura 7 (a) Liga de Simulação 2D; Figura 7 (b) Liga de Simulação 3D. O ambiente de simulação é baseado no sistema de simulação de utilização livre "*soccer server*". Este sistema, apresentado na Figura 7 (a), simula um campo contendo 2 (duas) equipes e uma bola, no qual se realiza um jogo virtual 2D de futebol. Cada equipe é composta de onze jogadores (chamados de agentes autônomos) e, eventualmente, 1 (um) treinador, que se conectam ao simulador, em uma arquitetura cliente-servidor, através de sockets UDP (*User Datagram Protocol*). Simulador é representado por um campo de futebol bidimensional. Este campo de futebol fica armazenado em um computador servidor. O servidor detém informações sobre a partida como posições dos jogadores e bola. O simulador aceita comandos de baixo-nível dos jogadores. Os jogadores possuem sensores (visuais, sonoros e físicos) e podem executar alguns comandos básicos (chute, interceptação, drible, etc.) com objetivo de influenciar no ambiente (Robocup, 2016; Stone, 2016).

A Liga de Simulação 3D, Figura 7 (b), conta com as mesmas características da 2D e acrescenta uma dimensão física extra ao jogo. Essa dimensão proporciona maior realismo no ambiente simulado, uma vez que cada jogador apresenta características humanoides. Novos comportamentos foram incorporados como abaixar e levantar (Robocup, 2016).

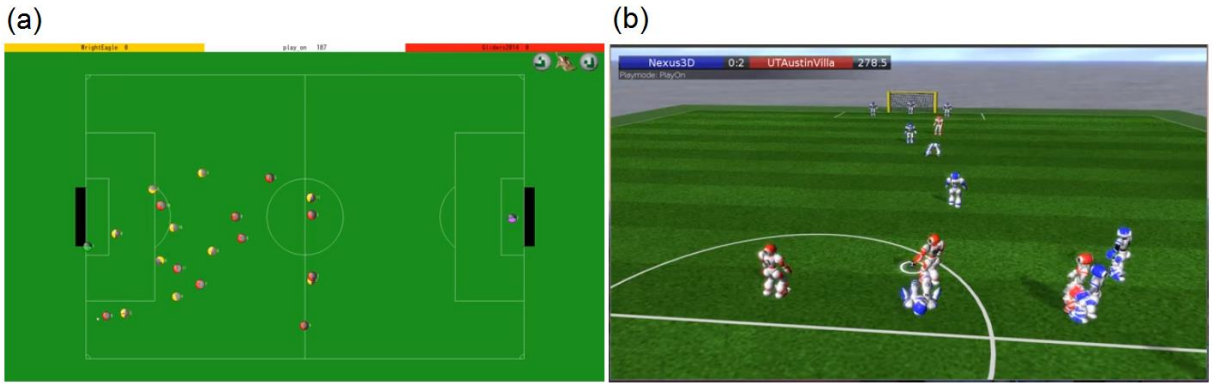


Figura 7 - Liga de simulação 2D/3D da RoboCup (Robocup, 2016).

[8] http://www.robocup2016.org

Ceebot

Ceebot é um jogo comercial com foco no aprendizado de programação de modo divertido. Ceebot utiliza linguagens de programação similares à C++, C# e Java, com a proposta da inclusão de conceitos pedagógicos alinhado com o desenvolvimento da criatividade. A Figura 8 exibe a interface do jogo (Epistec, 2015). Ceboot destaca-se na proposta de exercícios progressivos que remetem aos conceitos (variáveis, condicionais, classes, objetos, etc.) presentes nas ementas de ensino encontradas em disciplinas tradicionais de programação.

Figura 8 - Editor e Depurador do Ceebot (Epistec, 2015).

Furbot

O Furbot propõe exploração do raciocínio lógico e dos conhecimentos que o aluno detém da linguagem de programação. O jogo trabalha a programação lógica virtual de um robô em um ambiente bidimensional com interação com outros objetos. Esta interface permite ao aluno desenvolver códigos de movimentação com objetivo de transpor obstáculos propostos durante o jogo (Furbot, 2015). A Figura 9 exibe o *framework* do Furbot com um traçado a representar uma determinada missão a cumprir.

No jogo, o robô é estendido de uma classe e o aluno tem como objetivo programar a lógica do robô em um método denominado *inteligencia()*. Partes da programação do Furbot são expressas no idioma português. Na Figura 10 é possível observar os comandos: “andarDireita”; “ehVazio” e “diga” na qual o robô pode: andar uma célula de cada vez; visualizar as células

vizinhas; falar (envio de mensagens ao estudante); capturar objetos na vizinhança e adicionar/remover objetos no mundo.

O jogo Furbot possui, além da interface desktop, propriedades de execução em dispositivos móveis compatíveis com a plataforma Java Micro Edition (JavaMe). O Furbot alinha-se aos novos paradigmas da aprendizagem baseada em jogos com dispositivos móveis (*Mobile Game Based Learning* – MGBL) (Coelho, 2013; Furbot, 2015)

Figura 9 - Ambiente Furbot para Android (Coelho, 2013).

```
import br.furb.furbot.Furbot;
public class Exemplo01 extends Furbot {
      public void inteligencia() {
      while (!ehFim(DIREITA)) { //desloca o robô até limite direito do mundo
            if (!ehVazio(DIREITA)) //se não houver objeto no caminho
                  andarDireita(); //deslocar a próxima célula
            else;}}} // trata situação de obstáculo na direção do robô
```

Figura 10 - Programação da Inteligência do Furbot (Furbot, 2015).

WuCastle

O WuCastle é um jogo desenvolvido na Universidade da Carolina do Norte (Charlotte) para ensinar matrizes e laços (*loops*) de modo interativo e visual na linguagem de programação C++. Em um castelo como ambiente, a partir do personagem representado por uma fada com habilidades de programação, os alunos têm a possibilidade de programar bonecos de neve. O jogo permite aos desenvolvedores criar o ambiente com base em um mapa com opções de arrastar e soltar e edição de eventos. O WuCastle é um jogo de RPG (*Role-Playing Game*) no qual jogadores desempenham o papel de um personagem, que permite ao estudante desenvolver sua programação de modo interativo com exércitos de bonecos de neve. As saídas das jogadas são exibidas *on-line* para apoiar os jogadores na visualização dos códigos (Eagle & Barnes, 2008).

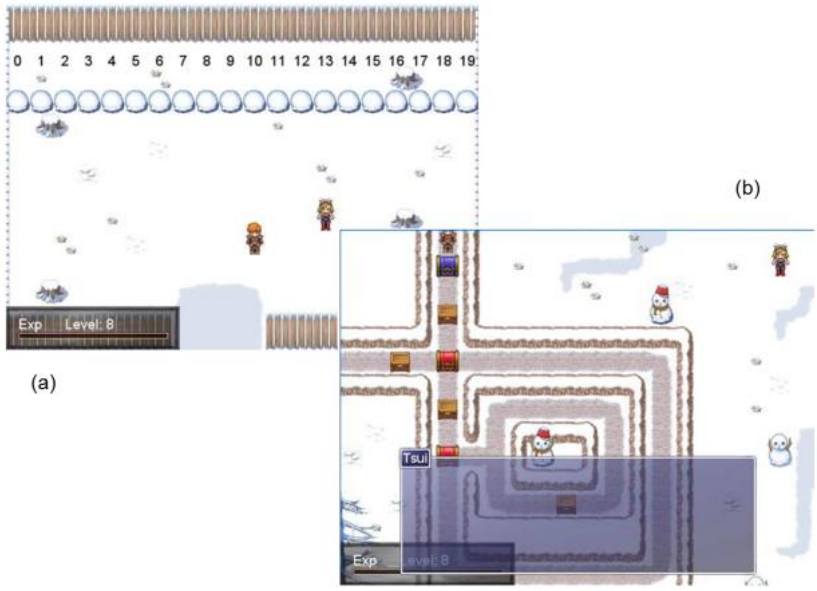


Figura 11 - Ambiente WuCastle (Eagle & Barnes, 2008).

É possível observar na Figura 11 que o jogo possui dois tipos principais de interação: (a) Mapa para manipular matrizes; (b) Ambiente do jogo para explorar o cenário com comandos animados de movimentação.

Scratch

Scratch é um *software* desenvolvido no MIT (Massachusetts Institute of Technology) Media Lab para ensinar lógica de programação com bases construcionistas. O Scratch possui, em todas versões atuais, suporte para o idioma português do Brasil. No ambiente do Scratch é

possível criar histórias interativas, desenvolver jogos, elaborar animações (com a recursos de importação de variados tipos de mídias: imagens, sons, músicas). Existe uma comunidade Scratch que permite aos usuários compartilhar e trocar informações acerca dos projetos (Scratch, 2015). A interface do jogo é constituída por blocos de comandos que devem ser arrastados e encaixados uns aos outros como peças de encaixar. Na combinação das peças (representando comandos), é possível formar programas sintaticamente corretos. Neste contexto, é possível que o usuário foque na lógica de programação abstraindo a sintaxe (Maloney, Resnick, & Rusk, 2010).

O público-alvo do Scratch é amplo. Existem referências na literatura que exibem a aplicação no ensino de programação para crianças do ensino fundamental até dinâmicas em cursos superiores de Ciência da Computação (Aureliano & Tedesco, 2012; Oliveira, Souza, Barbosa, & Barreiros, 2014). O Scracth é um *software* de programação que funciona como um facilitador utilizado em atividades com a necessidade de exploração individual ou colaborativa (Aureliano & Tedesco, 2012). A Figura 12 (a) apresenta um programa desenvolvido no ambiente Scracth, (b) seu equivalente em linguagem de programação Java e (c) o resultado final da programação no ambiente.

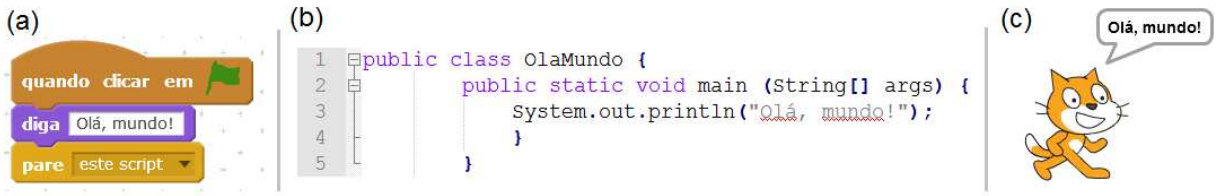


Figura 12 - Sintaxe Desenvolvida em Ambiente Scratch.

Woot!Craft

O Woot!Craft® é um curso para ensinar programação com Minecraft. Minecraft é um jogo independente, do tipo mundo aberto, escrito na linguagem de programação Java. O termo mundo aberto se equipara aos termos "*open world*" ou "*sandbox*", conceitos presentes em determinados jogos digitais em que são inseridas apenas limitações mínimas ao personagem, permitindo ao jogador explorar, modificar e selecionar livremente as tarefas que serão realizadas. No Minecraft, é possível construir estruturas utilizando cubos texturizados com várias opções de materiais: pedra, terra, areia, arenito etc. Além da coleta de recursos, o jogo

pode misturar sobrevivência e exploração. O *software* oferece ao jogador oportunidades criativas na construção de cidades ou civilizações (Cheverton, 2014).

O Woot!Craft® proporciona aprendizado *on-line* de linguagem de programação com base no Minecraft. Elaborado em forma de curso dentro do jogo, foi projetado para que alunos aprendam de modo intuitivo, lúdico e imersivo. Aprendizagem imersiva em jogos digitais implica no uso de elementos gráficos para replicar ambientes reais em um mundo virtual. Estudos propostos por Haycok apontam a relevância do tema aprendizagem imersiva para motivação dos alunos (Haycock, 2007). A estrutura modular do curso Woot!Craft permite a promoção de objetivos e estratégias que geram saídas em forma de pistas a serem usadas como fonte para elaboração da lógica e programação da sintaxe, conforme aumentam os níveis do jogo (denominados módulos) (Woot!Craft, 2015).

A proposta do Woot!Craft® em utilizar um jogo de base para implementar um curso, surgiu a partir da identificação do Minecraft como um dos jogos mais difundidos atualmente no mundo. Minecraft tem, atualmente, 10 (dez) recordes no Guinness Book dentre eles: (1) *Game* independente mais vendido; (2) Projeto Minecraft com maior número de downloads; (3) Primeiro país construído em escala real em um videogame (4) Maior número de jogadores em um mesmo mundo de Minecraft; (5) Filme de fã mais assistido baseado em videogames; (6) *Game* mais jogado na Xbox Live; (7) Beta mais popular; (8) Maior maratona em Minecraft; (9); Maior local real criado em Minecraft (foi criado mapa da Grã-Bretanha com suas ilhas em 22 bilhões de blocos); (10) Maior convenção de um jogo independente (Guinness, 2014).

No Woot!Craft o aluno aprende programação em missões progressivas realizadas no universo virtual do Minecraft. A temática do Woot!Craft se passa em um mundo povoado por robôs e computadores, no qual, para avançar de módulo, é necessário apender linguagem básica de programação. Por se tratar de um curso, o mesmo é regido por duração em horas (16 horas/aula). Os módulos apresentam o seguinte plano de curso: (1) Apresentação e conceitos básicos; (2) Introdução à linguagem de programação com *Blockly*: antes de jogar Minecraft aluno aprende conceitos do Google® *Blocky*, similar ao Scratch com programação de comandos em blocos de montar (Figura 13); (3) Programação com o Minecraft (Woot!Craft, 2015). A Figura 14 exibe cenários que podem ser utilizados para aplicação do curso Woot!Craft®.

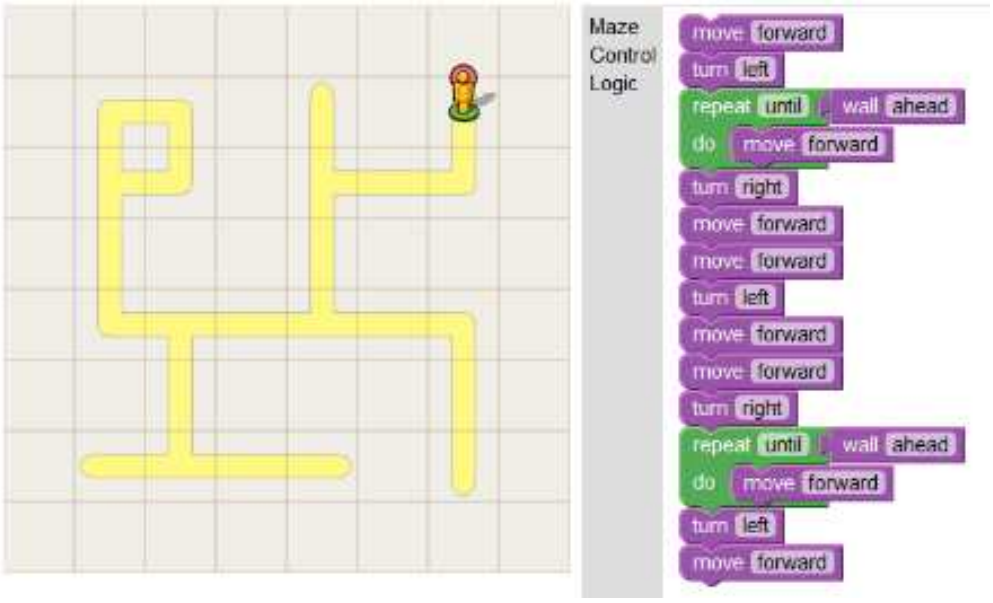


Figura 13 - Programação com Google Blockly.

Figura 14 - Cenários Minecraft com Possíveis Situações Woot!Craft.

3.3. *Framework* de Simulação "Robocode"

O projeto Robocode é um *software Open Source*, com fontes abertas e sob os termos da EPL (2016) (*Eclipse Public License*), que possibilita simulações de disputas com robôs virtuais com base nas linguagens de programação Java e C# (EPL, 2016; Fayek & Farag, 2015; Robocode, 2013). O Robocode foi projetado para ser um ambiente de simulação de robôs em uma arena, com o objetivo de comparar estratégias de programação com disputas estruturadas, tendo cada robô seus próprios algoritmos e comportamentos. A Figura 15 exibe uma típica batalha entre duas equipes de robôs no Robocode.

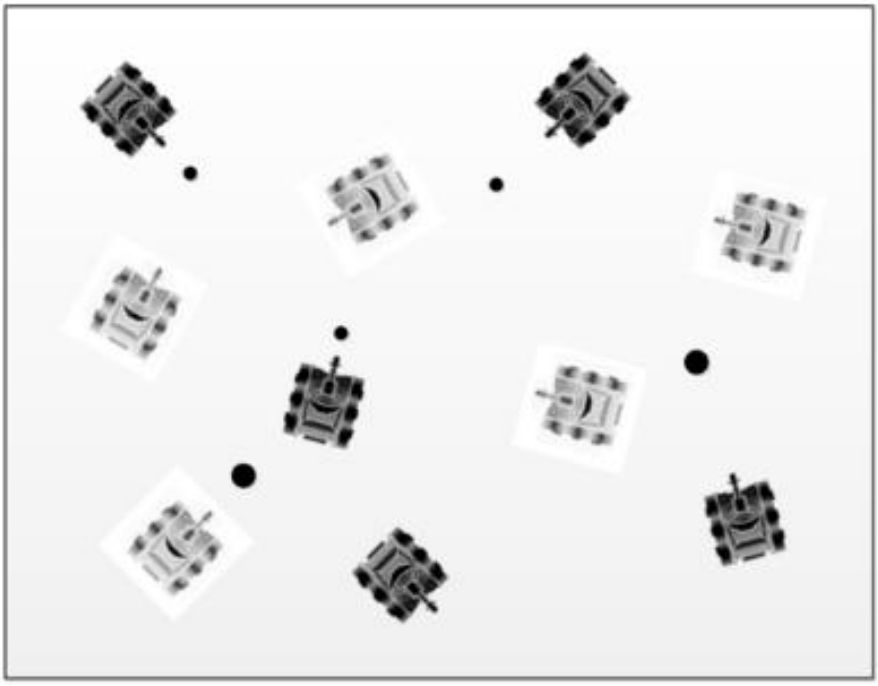


Figura 15 - Arena Robocode com duas equipes (Recchia, Chung, & Pochiraju, 2014).

Robocode foi desenvolvido por uma comunidade da IBM® denominada *AlphaWorks* (integrado ao centro de desenvolvimento *DeveloperWorks*), ativa desde 1996 com propostas de desenvolvimento de tecnologias emergentes. No início do ano de 2000, o projeto Robocode foi idealizado por Matthew A. Nelson com objetivo de desenvolver um simulador didático em tempo real. Além da proposta de um jogo educacional, Matthew queria quebrar paradigmas da época que classificavam as sentenças escritas para jogos em Java como lentas. O simulador Robocode nasceu com o lema "*Build the best - destroy the rest!* " (Li, 2002; Robocode, 2013).

Robocode ganhou notoriedade nos laboratórios de pesquisa da IBM®, porém com incertezas de sua continuidade, o projeto foi levado em 2005 para o *SourceForge*[9]. Em 2006, Flemming N. Larsen assumiu como administrador e desenvolvedor do projeto Robocode. No ano de 2010, a partir da versão Robocode 1.7.2, Pavel Savara desenvolveu um *plug-in* que ofereceu suporte à linguagem de programação C# (plataforma .NET). Atualmente, o projeto Robocode é mantido por iniciativas de *software* livre em torno da comunidade Robocode. A Figura 16 exibe a evolução do Robocode desde a criação do projeto.

[9] Repositório de código fonte que atua como centro para desenvolvedores gerenciarem projetos livres e de código aberto colaborativamente. *SourceForge* atua como um sistema de gerenciamento de repositórios de códigos-fonte. Foi o primeiro sistema de gerenciamento a fornecer esse tipo de serviço à projetos abertos.

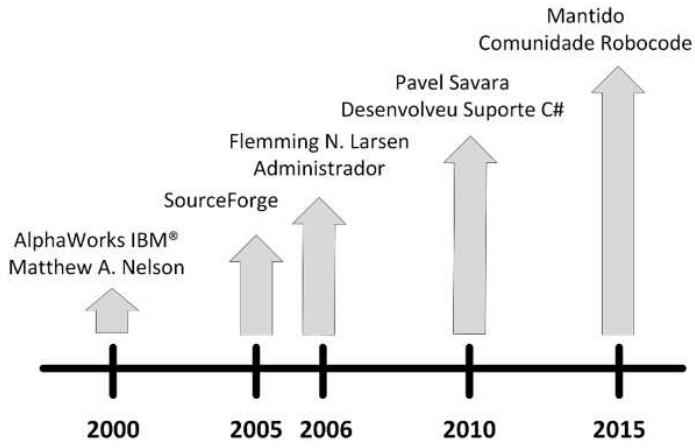


Figura 16 - Linha Evolução Robocode.

Pode-se classificar Robocode como um jogo sério (*Serious Games* - SGs), por ser um jogo que proporciona um ambiente em que o agente de aprendizagem pode interagir e aprender. No Robocode, o agente tem a possibilidade de escolher uma variedade de comportamentos e medir a eficácia de cada um ao longo do desafio (Nidorf, Barone, & French, 2010).

Robocode pode ser enquadrado como um *framewok* de simulação de fácil utilização originalmente criado para ensinar Programação Orientada à Objetos (POO) (Alaiba & Rotaru, 2008). A classificação do Robocode está centrada em um simulador de batalhas com base em Inteligência Artificial (IA) (Recchia et al., 2014). No simulador de disputas, é possível observar o comportamento dos robôs. Pelo fato das disputas ocorrerem de modo *on-line* na plataforma, é possível comparar a programação de robôs efetuada por outros desenvolvedores. No simulador, os robôs são entidades autônomas e os programadores não têm controle das ações, o que contribui para programação de agentes inteligentes. Um robô não possui total conhecimento do meio ambiente de batalha e suas informações estão restritas aos sensores do qual ele pode ler (ex. posição do robô, ângulo do canhão e/ou radar etc.) (Fayek & Farag, 2015; Hong & Cho, 2004).

Conforme visto, Robocode pode ser classificado conforme várias visões diferentes. A Figura 17 indica várias definições e enquadramentos para as variadas características de diferentes autores para o Robocode (Alaiba & Rotaru, 2008; Bonakdarian & White, 2004; Fayek & Farag, 2015; Hong & Cho, 2004; Li, 2002; Nidorf et al., 2010; Papert, 1993; Recchia et al., 2014; Robocode, 2013; Stahl, 1990; Stallman, 2014; Susi et al., 2007)

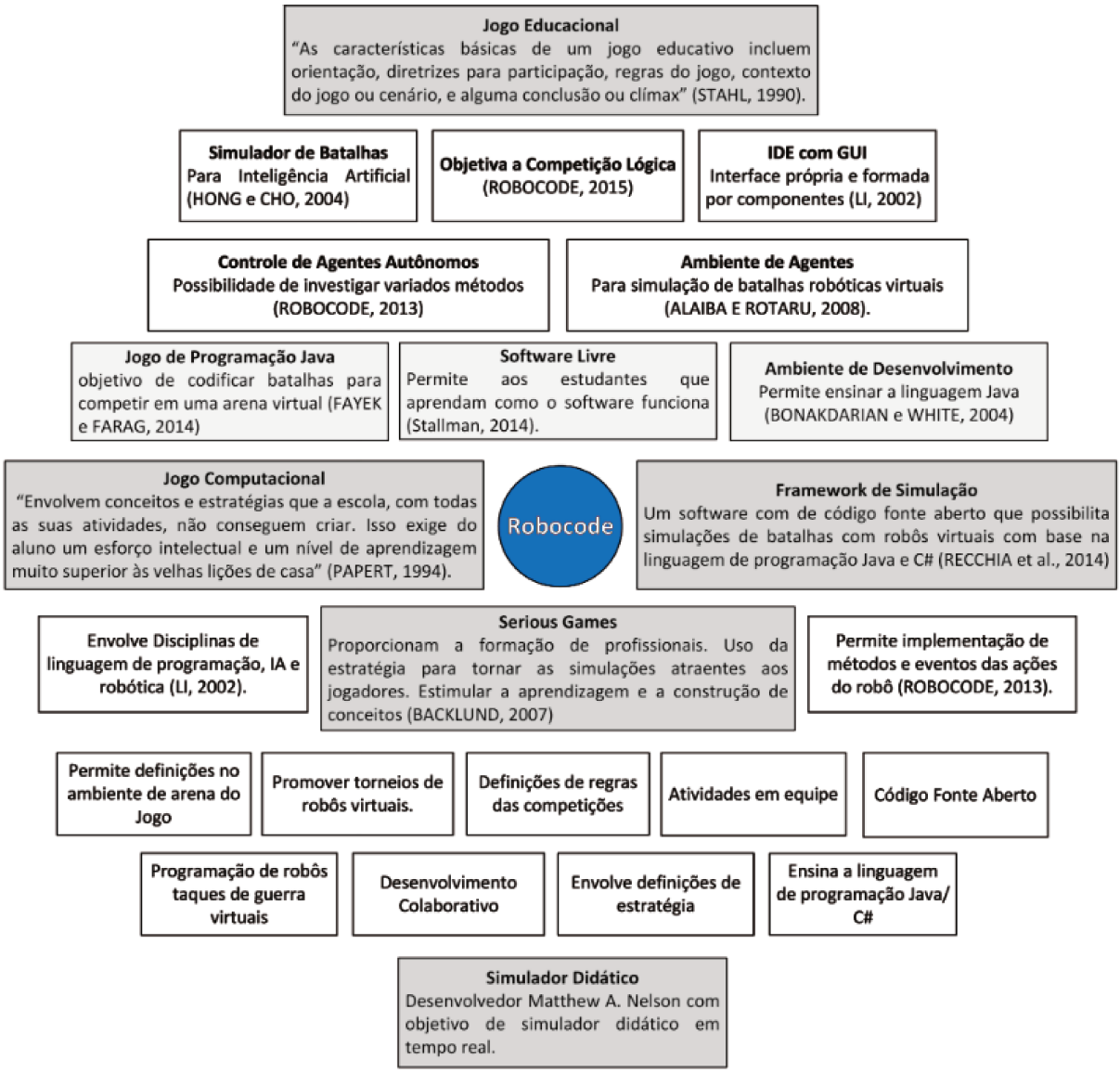


Figura 17 – Definições para Robocode.

3.3.1. Programação no Robocode

Robocode consiste em um jogo de programação com objetivo de codificar disputas para competir em uma arena com outros robôs virtuais (*Robot code* consiste na abreviação Robocode). O jogador é o programador do robô virtual, cuja função é desenvolver sua lógica com informações de comportamentos e reações a eventos que ocorrem na arena do jogo. Robocode é composto por um ambiente completo de desenvolvimento: instalador próprio, editor de robôs e compilador Java (Robocode, 2013).

Os robôs do Robocode têm a forma de tanques de guerra e são "treinados" por um agente programado. Os agentes são dotados de uma grande variedade de comportamentos disponíveis (classes e métodos próprios à plataforma) e dados sensoriais (localização, distância, energia, etc.). Conforme a evolução complexidade dos agentes, os tanques podem se tornar mais competitivos, assumindo comportamentos baseados em métodos estatísticos para analisar uma determinada situação e um campo de disputa e tomar decisões.

Desenvolvedores iniciantes a avançados podem criar projetos no Robocode. Para que os desenvolvedores possam aprender a criar os códigos, existem documentos da *Javadocs* API disponíveis no repositório *SourceForge* do Robocode. Programadores podem se divertir e treinar suas habilidades no desafio global chamado *best-of-breed* em variados torneios espalhados ao redor do mundo (Li, 2002).

Para programar um robô, usando Java, um jogador precisa derivá-lo de uma classe denominada "*robot*". A classe *robot* apresenta a estrutura do Robocode: veículo; canhão e radar, cada um podendo ser configurado de modo independente. No veículo, é possível determinar avanço, retrocesso e a direção do movimento. Acelerações e desacelerações lineares juntamente com as taxas de rotação podem também ser aplicadas junto ao movimento do veículo. O ambiente oferece a possibilidade de investigar métodos de controle de agentes autônomos. Permite o desenvolvimento de robôs autônomos simulados, da qual o programador define os critérios da "inteligência" para que os robôs operem disputas virtuais (Robocode, 2013).

Robocode consiste em um *software* com ambiente de agentes para simulação de batalhas robóticas virtuais. Há uma grande e consolidada comunidade desde 2003, denominada RoboWiki[10], de código fonte aberto em torno do ambiente, com a promoção de competições ao redor do mundo. O Robocode pode ser executado em plataformas com suporte às versões recentes do Java Runtime (versão 2 ou superior) (Alaiba & Rotaru, 2008; Robocode, 2013).

> A ideia básica por trás do Robocode é desenvolver *softwares* de agentes autônomos (robôs) para competir em um ambiente virtual fechado uns contra os outros. Cada robô é programado em Java e deve implementar uma determinada classe base. Este código segue uma arquitetura orientada a eventos do qual comanda a movimentação, localização de adversários e modo de disparo do robôs virtuais (Alaiba & Rotaru, 2008).

[10] http://robowiki.net

Li (2002), do *DeveloperWorks* da IBM®, indica a possibilidade de aprender herança, polimorfismo, manipulação de eventos e classes internas com o Robocode. Todos esses conteúdos, que fazem parte da ementa das disciplinas (ex. linguagem de programação) dos cursos na área de informática, podem ser trabalhados concomitante dentro do ambiente Robocode, em desafios que consistem em como evitar ser atingido por disparos ou executar manobras para uma intervenção de precisão (evitar bater na parede ou em robôs adversários), por exemplo. Um conjunto de robôs, desenvolvidos para servirem de testes para os desafios, fazem parte da biblioteca de exemplos do ambiente. O desenvolvedor pode criar projetos e participar em ligas e torneios.

O Robocode possui uma IDE (*Integrated Development Environment*) própria e uma interface formada por componentes GUI (*Graphical User Interface*), também conhecido por *widgets* (controles para simplificar o acesso a um outro programa ou sistema). A Figura 18 indica os componentes do sistema Robocode, na qual é possível visualizar as janelas de arena de disputa (*Battlefield*) e o editor do robô (*Robot Editor*), que representam a forma do IDE (Li, 2002).

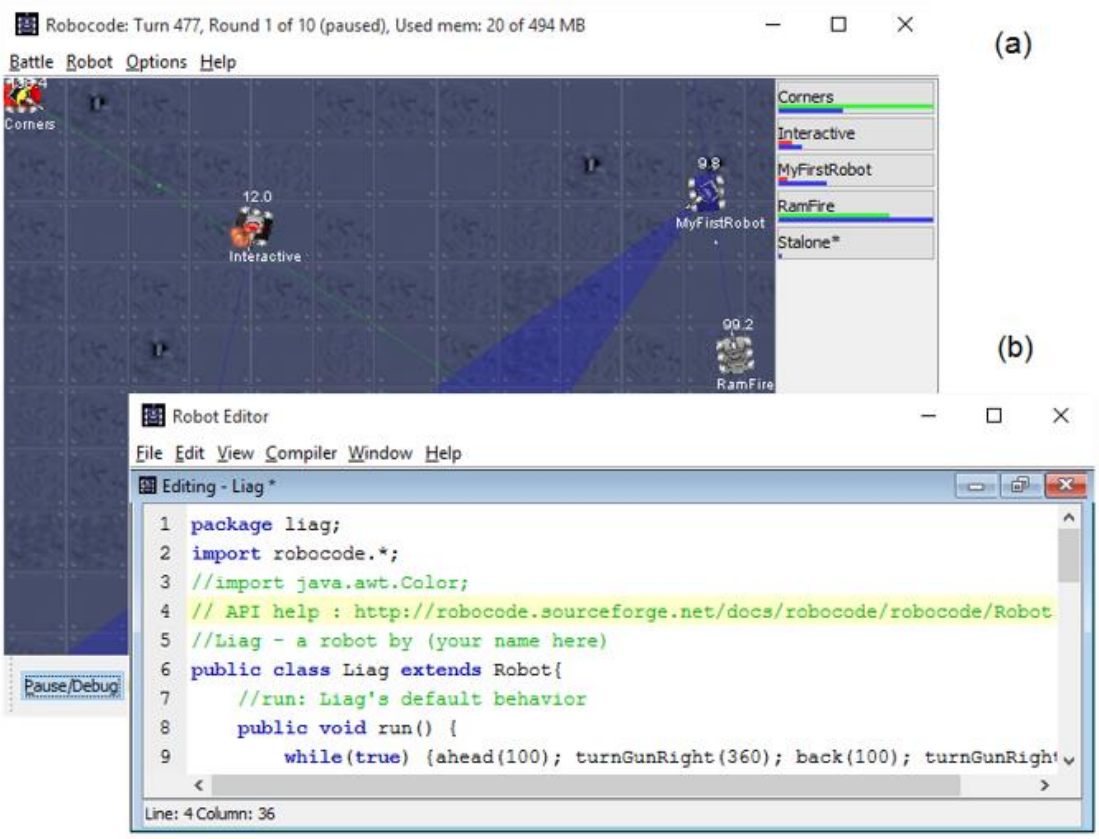


Figura 18 - Sistema Robocode (a) Arena de Disputas e (b) Editor de Robô.

O mecanismo principal de simulação, arena de disputas, é constituído por comandos que permitem criar, salvar e abrir batalhas existentes. Na arena é possível obter dados estatísticos dos robôs envolvidos ou mesmo pausar o jogo com comandos de fácil acesso (Li, 2002).

O editor de robôs é caracterizado por um compilador que permite aos desenvolvedores criarem e compilarem os projetos programados na linguagem Java ou C#. Qualquer robô criado e compilado com sucesso com o editor de robôs está pronto para participação em disputas. "Um robô no Rococode consiste em uma ou mais classes Java que podem ser arquivadas em um pacote JAR (Java ARchive) " (Li, 2002).

O tamanho da arena no Robocode pode ser configurado apenas antes do início das disputas. Como observado na Figura 19, cada robô possui suas próprias tarefas em uma fila de eventos. Além disso, cada robô possui o método de execução *run()*, no qual ocorrem os principais ciclos de processos. Grande parte da estratégia geral dos robôs é implementada neste método, o que diz ao robô virtual o que deve realizar enquanto não ocorrem eventos (Alaiba & Rotaru, 2008).

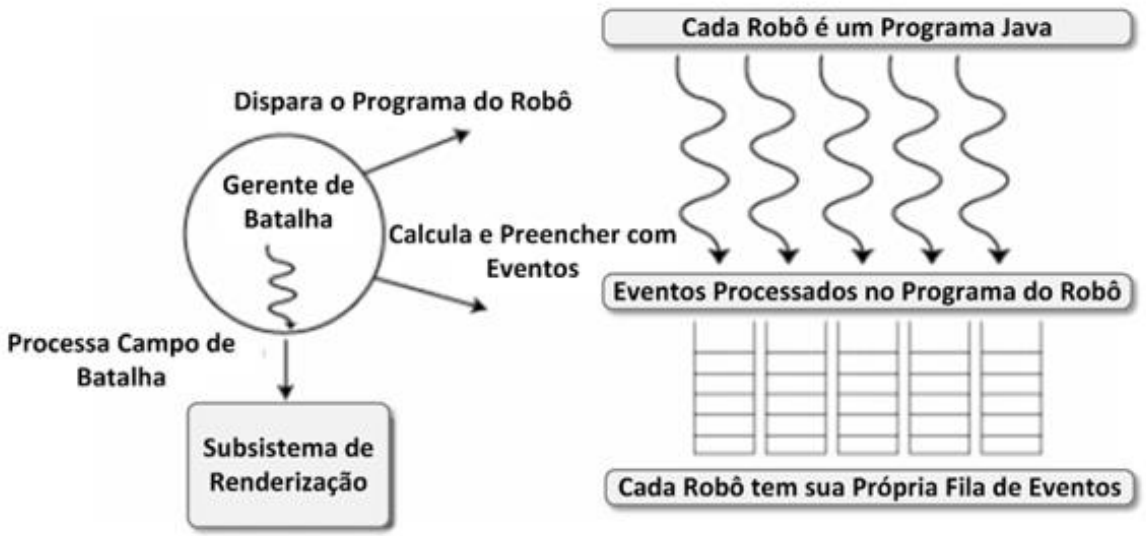


Figura 19 - Arquitetura do Robocode com mecanismo de simulação (Li, 2002).

O Robocode utiliza a estrutura de coordenadas cartesianas posicionada ao lado inferior esquerdo de origem (0,0). A altura (*Battlefield_heigh*) e largura (*Battlefield_width*) é configurada antes do início da batalha com atribuição de valores em pixels (RoboWiki, 2015). A Figura 20 exibe em sentido horário a conversão de 0/360 (graus) da orientação do jogo.

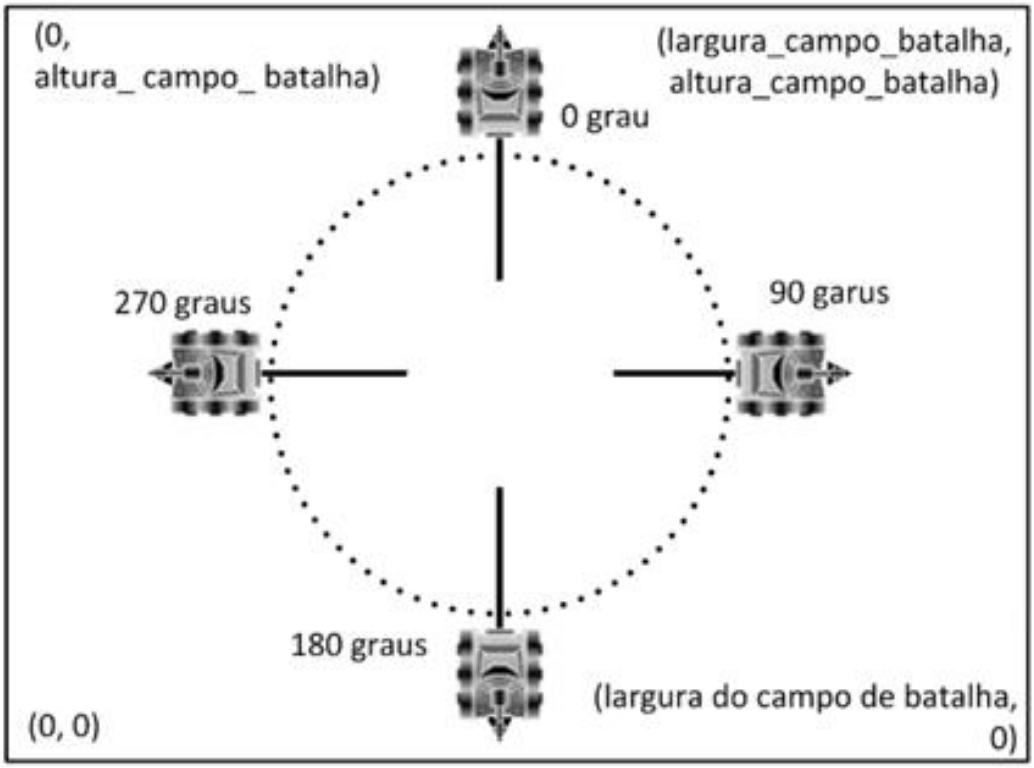


Figura 20 - Orientação Robocode, 0º Norte, 90º Oriente, 180º Sul, 270º Ocidente.

Cada robô é composto por um corpo (veículo), um canhão e um radar. O corpo é responsável pelo movimento do tanque e carrega o canhão e o radar. O canhão é utilizado para disparar: cada bala tem uma velocidade e consome uma quantidade de energia do robô de acordo com sua força. Quando uma bala atinge um adversário, o robô que efetuou o disparo ganha uma quantidade de energia proporcional aos danos causados. O ganho de energia ao atingir um adversário é calculado com base na potência do disparo (ex. robô dispara com potência "*X*" e perde "*X*" pontos da consequência do disparo, se atingir o adversário, ganha 3 vezes "*X*" pontos e o adversário perde 4 vezes "*X*" pontos). O radar é utilizado para localizar outros robôs adversários e obter suas informações. Cada parte possui suas funções operacionais e podem trabalhar de modo independente. A Figura 21 exibe a anatomia de um robô: (w) indica a direção do veículo; (x) eixo do movimento do veículo (*ahead* e *back*); (y) posição de disparo, *fire()*, seguido de direção e velocidade; (z) direção do canhão em graus ou radianos (*turnGunRight* e *turnGunLeft*) (Fayek & Farag, 2015; Li, 2002).

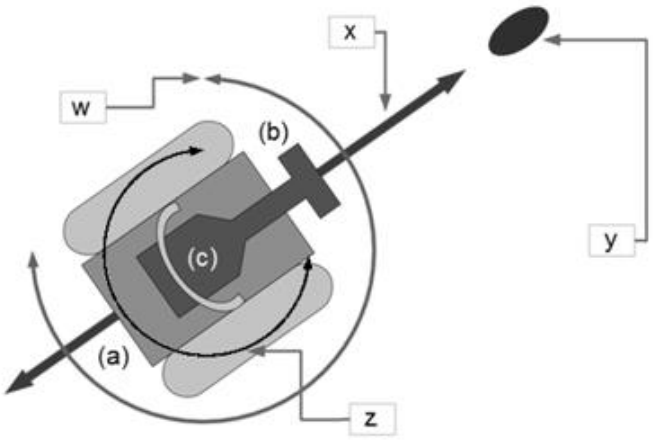


Figura 21 - Anatomia de robô no Robocode (a) veículo, (b) canhão e (c) radar (Li, 2002).

Um robô possui um método principal, e até outros cinco métodos em execução. O segmento principal geralmente tem um ciclo infinito de repetições no qual as ações são tomadas. Os códigos do Robocode fornecem determinados eventos para rastrear e captar informações do meio: como detecção de um robô adversário; quando o robô for atingido por uma bala; quando robô bater na parede; etc. (Hong & Cho, 2004; Robocode, 2013). A Figura 22 exibe o código base para programação básica de robôs. É possível observar os seguintes métodos e eventos: (1) evento *run()*, considerado o principal é iniciado quando começa um novo turno (*round*); (2) eventos como *onScannedRobot()*, executado quando encontra um adversário; (3) métodos como *ahead(x)*, para movimentação para frente onde a distância x é dada por parâmetro. O Apêndice I apresenta as ações detalhadas de programação do robô.

```
package stallone;
import robocode.*;
public class Stallone extends Robot{
      public void run() {
            while(true) {
                  ahead(100);
                  turnGunRight(360);
                  back(100);
                  turnGunRight(360);}
                }
      public void onScannedRobot(ScannedRobotEvent e) {
            fire(1);}
      public void onHitByBullet(HitByBulletEvent e) {
            back(10);}
      public void onHitWall(HitWallEvent e) {
            back(20);}
                }
```

Figura 22 - Código Base para Programação no Robocode.

3.3.2. Regras do Jogo e *Gameplay*

Frequentemente utilizado por *sites* e revistas especializadas em jogos, o termo *gameplay* muitas vezes aparece traduzido por "jogabilidade", termo ainda não encontrado em dicionários de língua portuguesa. O presente trabalho utiliza o termo *gameplay* como sendo "experiências que o jogador tem durante as interações com jogos digitais" (Howland, 1999).

Na *gameplay* do Robocode é possível inserir em uma arena 2D dois ou mais robôs virtuais em uma competição uns contra os outros. A competição é formada em turnos (*rounds*), onde cada robô é limitado a único tiro por unidade de tempo. Robôs começam com um nível de energia de 100 (cem) e perdem a "vida" quando este número atinge 0 (zero). O jogo termina quando restar apenas um único robô ou quando se esgota o limite de tempo do *round*. Os robôs podem se mover em mais da metade da velocidade máxima da bala.

O objetivo do robô é sobreviver frente a seus adversários durante a disputa. Um robô possui várias informações de si e de seus adversários para decidir o comportamento com base na coleta de dados ao longo da partida. Jogadores têm a função de analisar a complexidade das ações, com objetivo de desenvolver estratégias adequadas de comportamentos do robô para vencer o jogo. A forma de movimentação do adversário e uma eficiente utilização da energia devem ser considerados para o sucesso das estratégias de jogo (Hong & Cho, 2004).

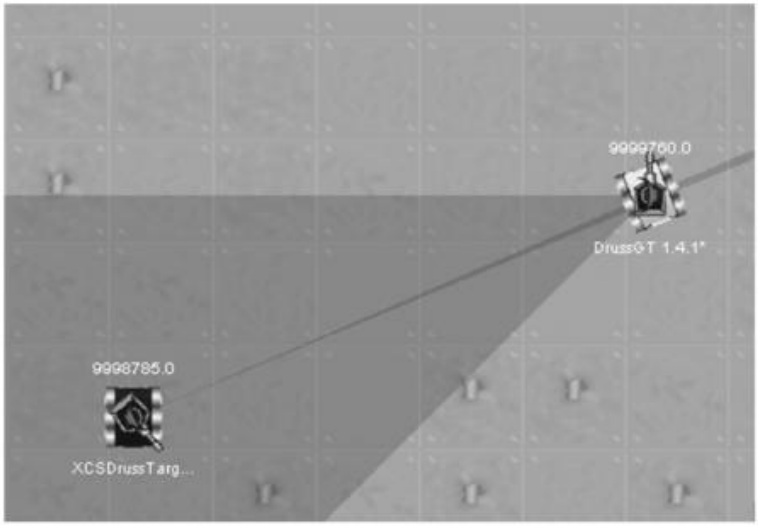


Figura 23 - Arcos Cinza Escuro Representam Área Detectada pelo Radar.

O robô pode se movimentar a partir de controles de distância para frente ou trás, com as direções esquerda ou direita. Os robôs têm sua movimentação de modo simultâneo entre eles a

partir do início da disputa. Com objetivo de identificar adversários, os tanques virtuais são equipados com radares que podem identificar um arco de até 45º (graus) do campo da arena do jogo (Figura 23). "Após rastrear a posição do adversário (*heading*), são retornados valores da velocidade e nível de energia. O radar não retorna à localização dos tiros ou rotação do canhão do inimigo" (Nidorf et al., 2010). Detalhes específicos das regras do jogo e do *gameplay* são apresentados no Apêndice II.

3.3.3. Naval Robocode

Em 2015, foi lançado pelo SourceForge, um projeto denominado Naval Robocode. Este *software* usa a base da plataforma do ambiente Robocode para simular disputas entre navios. Os tanques foram substituídos por navios e o ambiente do jogo se passa no oceano conforme apresentado na Figura 24. Além de compartilhar a mesma plataforma, compartilham muitas regras presentes no Robocode.

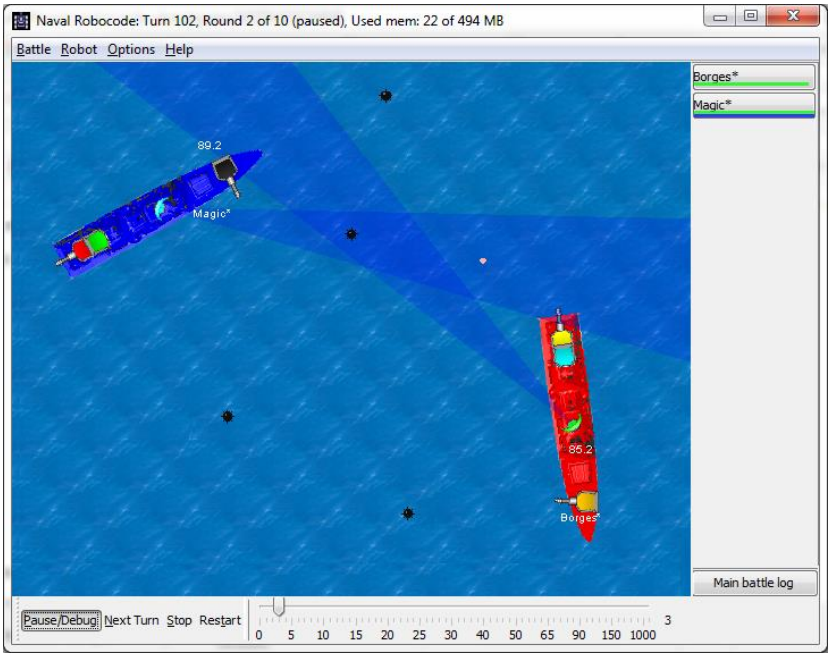


Figura 24 - Ambiente Naval Robocode.

Associado à estrutura do Robocode, o Naval Robocode incorpora novas características que adicionam novos fundamentos à dinâmica do jogo. Como características incorporadas: (1) No Robocode existe 1 (um) canhão enquanto no Naval Robcode são 2 (dois) canhões; (2) Os navios podem lançar minas. O fato de controlar 2 (dois) canhões e a possibilidade de lançar

minas, adicionam novas dinâmicas e possibilidades ao Naval Robocode para programação dos navios e definições das estratégias. Essas novas dinâmicas de jogo remetem a possibilidades de estratégias como efetuar disparos simultâneos a 2 (dois) adversários ou determinar meios de identificar e desviar de minas lançadas.

3.4. Aprendizagem em Competições Científicas de Programação

A presente seção exibe ambiente de competições científicas de programação para aprendizagem em empresas e áreas acadêmicas. Cada competição sugere seu objetivo final que pode ser a busca por talentos até o apoio na aprendizagem de conceitos de programação. A Olimpíada Internacional de Informática (*International Olympiad in Informatics* – IOI) consiste em uma das mais importantes e conhecidas competições de programação do mundo. No Brasil, a Sociedade Brasileira da Computação (SBC) se destaca por desenvolver práticas com competições científicas de programação com objetivo na popularização da computação nas instituições de ensino. A seguir, identificado por áreas, são apresentadas algumas das principais iniciativas em competições científicas de programação no Brasil e no mundo.

Setor Empresarial

Algumas empresas, como Facebook®, Google® e Hewlett Packard® (HP), mantêm competições científicas de programação para incentivar a inovação de produtos. Por exemplo, o botão "Curtir" do Facebook® foi criado a partir de *hackathons* internos. Outra iniciativa do Facebook® é a maratona anual mundial de programação denominada *Facebook Hacker Cup*[11]. *Hacker Cup* ocorre desde 2011 e está associada à busca de talentos que possam resolver algoritmos e inovar produtos (como a *Timeline* – Linha do Tempo – principal produto do Facebook). Há competições científicas de programação no Google, também com objetivo de identificar talentos, como exemplo, *Code Jam*[12]. O Google *Code Jam* ocorre desde de 2003 e busca profissionais de engenharia para emprego potencial no Google. Basicamente, a competição da *Google* consiste na resolução de problemas com envolvimento de algoritmos à

[11] https://www.facebook.com/hackercup
[12] https://code.google.com/codejam

serem resolvidos em uma determinada fatia de tempo (Briscoe & Mulligan, 2014; Digiampietri et al., 2014). O *HP CodeWars*[13] é um concurso anual de programação realizado desde 1997 pela empresa HP. A primeira competição foi realizada na cidade de Houston no Texas (EUA) e atualmente ocorre em diversas cidades espalhadas no mundo (Houston-EUA, Austin-EUA, Conway-EUA, Roseville-EUA, Palo Alto-EUA, Newcastle-ING, Barcelona-ESP, Bangalore-IND e Taipei-TAI). Voltado ao ensino médio, o *HP CodeWars* busca incentivar pesquisas com tecnologias nas escolas (Digiampietri et al., 2014).

Na América Latina, estudos mostram a organização de *hackathons* na empresa de origem argentina MercadoLibre®. MecadoLibre (no Brasil Mercado Livre) consiste em uma empresa de desenvolvimento de tecnologias e soluções para comércio eletrônico. A empresa realiza anualmente maratonas de programação em eventos culturais para desenvolvedores. Na competição, os desenvolvedores são avaliados pela criatividade e adequação as propostas em criar *softwares* de venda utilizando a *Application Programming Interface* (API) do MercadoLibre. Tecnicamente, a partir do acesso em uma API *on-line* livre, programadores podem previamente estudar, desenvolver aplicações e no dia do evento *hackathon* apresentar suas soluções (MercadoLibre, 2016; Rodriguez, 2015).

No Brasil, a Embrapa[14] (Empresa Brasileira de Pesquisa Agropecuária) do Ministério da Agricultura, Pecuária e Abastecimento (MAPA), organiza *hackthons* com objetivo de valorizar e reconhecer a criatividade dos alunos em criar novas soluções baseadas em Tecnologia de Informação, com foco em *software*, que possam contribuir para o setor agrícola brasileiro. No *hackathon* da edição 2016, o objetivo consistiu no desenvolvimento de aplicativos móveis para tomada de decisão no manejo integrado de pragas, visando a sustentabilidade dos agrossistemas (Embrapa, 2016).

Na Área Acadêmica

Na área acadêmica, competições científicas com linguagem de programação e projetos de desenvolvimento de *software* estão consolidando-se como abordagens efetivas para o desenvolvimento de habilidades na área da computação. No mundo, é possível citar a competição de programação *International Olympiad in Informatics* (IOI), voltada para o público de estudantes do ensino médio e ingressantes em cursos superiores. Organizada desde

[13] http://www.hpcodewars.org
[14] https://www.embrapa.br/hackathon

1989, IOI tem por objetivo a promoção da Ciência da Computação entre jovens para descoberta de talentos e estímulo aos alunos na programação. Outra iniciativa de destaque é a ACM *International Collegiate Programming Contest*[15] (ACM-ICPC), com público de estudantes de graduação e ingressantes na pós-graduação. Na ACM-ICPC, equipes com máximo de 3 integrantes, trabalham na resolução de desafios de algoritmos de programação. Os níveis de dificuldades das tarefas variam conforme as categorias (graduação e pós-graduação) e de acordo com a priorização de tempo utilizado em cada tarefa (Hakulinen, 2011).

No Brasil, a Sociedade Brasileira da Computação (SBC) em parceria com Fundação Carlos Chagas, realiza o evento denominado "Maratona de Programação[16]". O evento surgiu em 1996 a partir de competições regionais classificatórias para finais do concurso de programação da ACM *International Collegiate Programming Contest*. A Maratona de programação consiste na regional sul americana da competição. O público-alvo da competição são alunos de graduação e pós-graduação da área de computação e afins. Tem por objetivo promover a criatividade, trabalho em equipes, desenvolvimento de novas soluções de *software* e habilidade de resolver problemas. As equipes formadas por 3 (três) alunos são selecionadas dentre várias instituições de ensino brasileiras para participar das finais mundiais (SBC, 2016).

Outro evento organizado pela SBC, em parceria com Instituto de Computação da UNICAMP, consiste na "Olimpíada Brasileira de Informática (OBI)[17]". OBI tem o objetivo de despertar nos alunos o interesse por uma ciência importante na formação básica atualmente (computação). Com envolvimento de desafios, engenhosidade e ludicidade em competições, a OBI também integra uma fase sul americana, assim como maratona de informática, para classificação na ACM (SBC, 2016).

As metodologias presentes nas maratonas de programação da SBC associam 3 (três) abordagens de aprendizado: cooperativa, competitiva e baseada em problemas. Nas maratonas, os alunos participam de equipes colaborativas visando bons resultados dentro do time. As equipes devem: promover discussões para compreensão do enunciado; possibilidades de resoluções; analisar aspectos técnicos computacionais para ampliar os conhecimentos do grupo (Piekarski et al., 2015).

[15] ACM *International Collegiate Programming Contest* consiste na competição anual de programação entre universidades do mundo todo.
[16] http://maratona.ime.usp.br
[17] http://olimpiada.ic.unicamp.br

4. MATERIAIS E MÉTODOS

O objetivo deste capítulo é descrever as técnicas, metodologias e ferramentas que foram utilizadas no desenvolvimento do trabalho. As seções a seguir exibem as técnicas de pesquisa usadas no trabalho bem como os métodos selecionados para desenvolvimento e aplicação na competição científica do Robocode.

4.1. Técnicas de pesquisa

Entrevistas foram conduzidas junto aos professores que atuaram no ambiente de aplicação do trabalho. As técnicas de documentação direta foram selecionadas como um conjunto de preceitos e processos de que se serve uma ciência (Marconi & Lakatos, 2010). Para a técnica de entrevista, o trabalho trata a observação direta intensiva na qual "caracteriza como um encontro entre duas pessoas, a fim de que uma delas obtenha informações a respeito de um determinado assunto" (Marconi & Lakatos, 2010). A entrevista tem como objetivos o planejamento da competição e orientação em questões para entender problemas gerados durante as disputas dentro da competição a serem associados com PBL e o ambiente do Robocode.

Observação direta do tipo pesquisa de campo foram realizadas em instituições previamente selecionadas com objetivo de coletar informações acerca de um problema, para o qual procura-se resposta, ou uma hipótese, que deseja-se comprovar. O tipo de pesquisa de campo selecionado foi o quantitativo-descritivo: com análise das características dos fatos; avaliação da competição e isolamento de variáveis de pesquisa (relacionadas as hipóteses). Existiu o emprego da coleta de dados sobre a população em pesquisa (alunos das instituições). A pesquisa de campo foi apoiada por questionários (Marconi & Lakatos, 2010).

A pesquisa-ação foi utilizada como um tipo de pesquisa social com base empírica que é concebida e realizada em "estreita associação com a ação ou com resolução de um problema coletivo e no qual os pesquisadores e os participantes representativos da situação ou do problema estão envolvidos de modo cooperativo ou participativo" (Thiollent, 2008). Nesta modalidade de pesquisa foram criados ambientes propícios para que os participantes possam conhecer os problemas, mobilizar-se em ações para solucionar problemas (Nunes & Infante,

1996). Os problemas foram identificados a partir do estudo da competição de programação Robocode do ano de 2015 (Liga LIAG Robocode).

Questionários foram adotados como fonte de dados relacionados anterior e posteriormente à competição científica Robocode Brasil 2016 como observação direta extensiva. Os questionários têm por objetivos: (1) Verificação da hipótese; (2) Avaliação em busca de efeitos e resultados da competição científica; (3) Características quantitativa-descritivas de populações (gêneros, organizações, atitudes, opiniões, etc.); (4) Descoberta de variáveis potencialmente relevantes à pesquisa. O questionário foi constituído por uma série ordenada de perguntas, que foram respondidas por escrito e sem a presença do entrevistador (Marconi & Lakatos, 2010).

As perguntas dos questionários foram classificadas dos seguintes modos com relação ao tipo: (1) Abertas – livres ou não limitadas, são as perguntas que permitem ao informante responder livremente, usando linguagem própria e emitir opiniões; (2) Perguntas fechadas ou dicotômicas – também denominadas de limitadas ou de alternativas fixas; (3) Múltipla escolha do tipo mostruário – respostas possíveis são estruturadas junto a uma pergunta, devendo ao informante assinalar uma ou várias delas (deve ser explicada quando se deseja uma só resposta); (4) Múltipla escolha do tipo avaliação ou estimação – consiste em emitir um julgamento em escala com vários graus de intensidade para um mesmo item. As respostas sugeridas são quantitativas e indicam um grau de intensidade crescente ou decrescente. As perguntas foram classificadas também quanto ao seu objetivo: (1) Perguntas de fato – diz respeito a questões concretas, tangíveis, fáceis de precisar, referem-se a dados objetivos: idade, sexo, profissão, domicílio, estado civil etc.; (2) Perguntas de ação – atitudes ou decisões tomadas pelo indivíduo; (3) Perguntas de opinião – representação da pesquisa (Marconi & Lakatos, 2010).

Para desenvolvimento dos questionários e coleta de informações adotou-se a ferramenta *on-line* de criação e gerenciamento de formulários da Google®. Canais oficiais, gerenciador da competição e redes sociais, do Robocode Brasil 2016 foram utilizados para compartilhar os questionários. Solução *on-line* TIBCO® SpotFire foi utilizada no auxílio ao desenvolvimento da análise e visualização a partir dos dados selecionados nos questionários com objetivo em descobrir tendências ou padrões. Para organizar e gerenciar artigos e referências bibliográficas, foi utilizado o *software* gratuito Mendeley® na modalidade *on-line* e *desktop*.

4.2. Competições Robocode e Métodos

Este trabalho buscou analisar a utilização do método de PBL em competições científicas com jogo educacional Robocode para motivar alunos no estudo da disciplina de linguagem de programação. PBL representa um método de ensino-aprendizagem centrado no aluno, com estratégias voltadas à resolução de problemas. Com o PBL, o estudante deixa de ser um receptor passivo do conhecimento e assume papel do agente principal responsável pelo aprendizado. A situação do professor também muda, não seguindo as estratégias do ensino instrucional, assumindo a função de facilitador na construção do conhecimento (Gil, 2008). O problema é o *design* de um robô, estando alinhado com abordagem LBD.

O trabalho analisa, no estudo de caso, as competições: Liga LIAG Robocode 2015 e Robocode Brasil 2016, ambas com jogo educacional Robocode. O Robocode possui um ambiente de desenvolvimento de programas em linguagem de programação. O ambiente do Robocode foi utilizado para resolução dos problemas reais propostos a partir do PBL. Com apresentação das propostas e definição dos problemas ao longo da competição, os alunos devem: (1) Ter acesso ao ambiente de desenvolvimento para programar os robôs; (2) Criar a partir de conhecimentos prévios e pesquisas estruturadas os códigos necessários; (3) Compilar e corrigir possíveis erros; (4) Inserir no Robocode para acompanhar o resultado e planejar ("treinar") o robô conforme as características das disputas do ambiente de competição.

O estudo de caso deste trabalho foi baseado em ação em parceria do Laboratório de Informática, Aprendizado e Gestão (LIAG[18]) da Faculdade de Tecnologia (FT) da Universidade Estadual de Campinas (UNICAMP) com o Instituto Federal de Educação Ciência e Tecnologia (IFSP[19]). As instituições de ensino que compuseram o ambiente da competição científica foram: FT-UNICAMP – Faculdade de Tecnologia (Limeira - SP); COTIL-UNICAMP - Colégio Técnico de Limeira (Limeira - SP); IFSP – Instituto Federal São Paulo (Capivari – SP); IFSP – Instituto Federal São Paulo (Hortolândia – SP); IF Farroupilha - Instituto Federal Farroupilha (Campus São Borja - RS); UNICAMP - Universidade Estadual de Campinas (Campinas – SP); COTUCA - Colégio Técnico de Campinas (Campinas - SP); ETEC - Escola Técnica Estadual Adolpho Berezin (Mongaguá – SP); Faculdade Anhanguera (Rio Claro - SP); Faculdade

[18] Laboratório da FT-UNICAMP com a missão de integrar os estudos nas áreas de informática, aprendizagem e gestão de modo a buscar resultados superiores e diferenciados nas três áreas.
[19] Autarquia Federal de ensino vinculada ao Ministério da Educação (MEC) fundada em 1909 e reconhecida pela sociedade paulista por sua excelência no ensino público gratuito de qualidade (IFSP, 2015).

Anhanguera (Santa Bárbara d'Oeste – SP); Faculdade SATC (Criciúma – SC); PUC - Pontifícia Universidade Católica do Paraná (Londrina – PR); UFBA - Universidade Federal da Bahia (Salvador – BA); UFMG - Universidade Federal de Minas Gerais (Belo Horizonte – MG).

Os dados das competições científicas avaliados no trabalho são referentes ao ano de 2015 e 2016. A competição científica proposta no ano de 2016 recebeu o nome de Robocode Brasil[20].

[20] Robocode Brasil, http://www.robocodebrasil.com.br, consiste no canal oficial da liga brasileira de Robocode. O ambiente gerenciador da competição permite: inscrições e agrupamentos das instituições; equipes e participantes; receber códigos e desempenhar as partidas; gerenciar os resultados e produzir relatórios de status; consulta as regras e edital da competição; canal de comunicação com *newsletter* e capacitações no Robocode.

5. AMBIENTE DA COMPETIÇÃO CIENTÍFICA ROBOCODE

A competição científica de programação foi baseada no ambiente Robocode. A competição desenvolvida no presente trabalho teve por objetivo auxiliar o aprendizado de conceitos e práticas de linguagem de programação. Atividades envolveram etapas distintas como a cooperação em grupo para definições de estratégias na programação do robô até um modo de ludicidade no momento de descontração das disputas. O presente capítulo apresenta competições do Robocode no mundo, no Brasil e caracteriza o ambiente de aplicação do presente trabalho na competição científica formada por Torneios descentralizados institucionais e uma Liga nacional. De modo a privilegiar e tornar mais simples a leitura deste texto, será adotado o termo "Robocode Brasil" para representar a competição nacional científica do ano de 2016 e o termo "Liga LIAG Robocode" para referenciar a competição do ano de 2015, ambas desenvolvidas e organizadas pelo laboratório LIAG da FT-UNICAMP. A equipe do LIAG responsável por organizar os eventos que envolveram o Robocode (em 2015 e 2016) foi denominada de equipe Administração da Competição.

5.1. Histórico e Competições no Mundo

Em 2002 foi criado, por Christian Schnell (pesquisador da Universidade Técnica de Berlim), um gerenciador da liga mundial Robocode denominado *RobotLeague*. O gerenciador tinha funções de garantir o agrupamento dos participantes; desempenhar as partidas; gerenciar os resultados e produzir relatórios de *status* em páginas HTML compartilhadas na internet. *RobotLeague* foi descontinuada um ano após sua criação, provavelmente devido às incertezas sobre a continuidade do projeto na IBM, porém ganhou notoriedade por representar a primeira iniciativa de liga mundial do Robocode (Li, 2002). Atualmente, destaca-se a competição mundial chamada *Hat League* mantida pela comunidade Robowiki associada a iniciativa colaborativa denominada *RoboRumble*[21] para exibir as partidas e construir rankings. A Figura 25 exibe a página inicial da iniciativa *RoboRumble*.

[21] RoboRumble (2016) Disponível em: <http://robowiki.net/wiki/RoboRumble>. Esforço colaborativo mantido pela RoboWiki com objetivo de gerenciar e manter rankings *on-line* atualizados das disputas mundias de robôs distribuídas em categorias do Robocode.

Figura 25 - Página inicial do gerenciador RoboRumble.

A Figura 26 (a) exibe o ranking mundial de pontuações nas disputas "um contra um" na *RoboRumble*. Entre 1121 participantes, o Brasil ocupa a 7ª colocação com 99,37% PWIN (*Percentage WIN* – Porcentagem de Vitórias) e 84,19% de APS (*Average Percentage Score* - Pontuação Média Percentual) em pontos com as disputas realizadas até a data de 30 de agosto de 2016 (nome do robô mn.Combat 3.23.1). Ainda na Figura 26 (b) é possível observar os detalhes do robô brasileiro, mn.Combat 3.23.1, representante na *RoboRumble* (*Bot* na *RoboRumble* consiste em acrônimo da palavra *Robot*). Infelizmente, não se conseguiu informações sobre os autores desse robô.

(a)

	Flag	Competitor	APS	-PWIN	Latest Battle
1		jk.mega.DrussGT 3.1.3	91.41	100.0	2016-08-30 12:50:47
2		voidious.Diamond 1.8.22	90.69	99.96	2016-08-24 21:06:49
3		lxx.Tomcat 3.68	88.49	99.82	2016-08-30 12:44:26
4		abc.Shadow 3.83c	85.45	99.64	2016-08-30 11:58:44
5		aw.Gilgalad 1.99.5c	88.62	99.46	2016-08-30 12:36:20
6		kc.serpent.WaveSerpent 2.11	87.56	99.46	2016-08-30 12:31:35
7		mn.Combat 3.23.1	83.44	99.37	2016-08-13 06:24:23
8		darkcanuck.Pris 0.92	85.43	99.29	2016-08-30 12:15:39
9		voidious.Dookious 1.573c	87.17	99.29	2016-08-30 12:36:39
10		xander.cat.XanderCat 12.8	87.45	99.2	2016-08-30 12:50:40
11		cs.Nene 1.0.5	87.99	99.02	2016-08-28 08:57:22
12		davidalves.Phoenix 1.02	86.66	98.93	2016-08-30 12:22:06

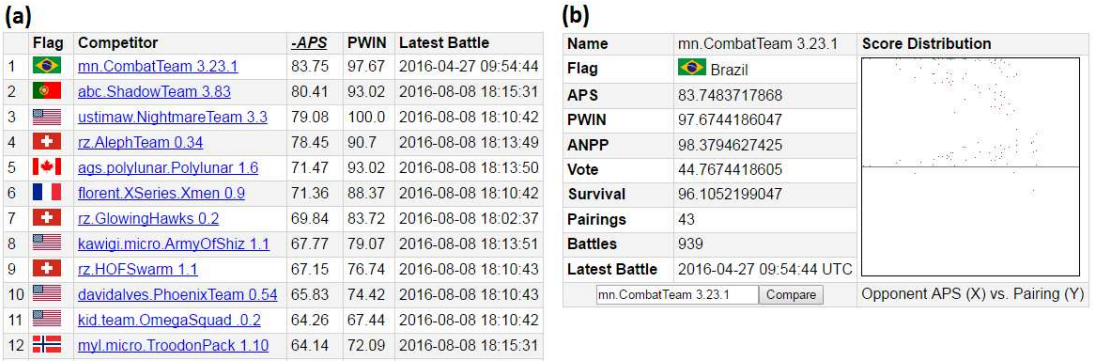
(b)

Name	mn.Combat 3.23.1	Score Distribution
Flag	Brazil	
APS	83.4404276172	
PWIN	99.373881932	
ANPP	88.0740972671	
Vote	0.109728549985	
Survival	93.510463913	
Pairings	1118	
Battles	16050	
Latest Battle		
2016-08-13 06:24:23 UTC		Opponent APS (X) vs. Pairing (Y)

Figura 26 - (a) Ranking RoboRumble "Um Contra Um" (b) Detalhes robô brasileiro.

A Figura 27 (a) exibe o ranking mundial de pontuações nas disputas da categoria "times" na *RoboRumble*. Entre 44 times participantes, o Brasil ocupa a 1ª colocação com 97,67% PWIN e 83,75% de APS em pontos nas disputas realizadas até a data de 27 de abril de 2016 (nome do

robô do time Brasil, mn.CombatTeam 3.23.1). Ainda na Figura 27 (b) é possível observar os detalhes da equipe brasileira, mn.CombatTeam 3.23.1.



(a)

	Flag	Competitor	-APS	PWIN	Latest Battle
1		mn.CombatTeam 3.23.1	83.75	97.67	2016-04-27 09:54:44
2		abc.ShadowTeam 3.83	80.41	93.02	2016-08-08 18:15:31
3		ustimaw.NightmareTeam 3.3	79.08	100.0	2016-08-08 18:10:42
4		rz.AlephTeam 0.34	78.45	90.7	2016-08-08 18:13:49
5		ags.polylunar.Polylunar 1.6	71.47	93.02	2016-08-08 18:13:50
6		florent.XSeries.Xmen 0.9	71.36	88.37	2016-08-08 18:10:42
7		rz.GlowingHawks 0.2	69.84	83.72	2016-08-08 18:02:37
8		kawigi.micro.ArmyOfShiz 1.1	67.77	79.07	2016-08-08 18:13:51
9		rz.HOFSwarm 1.1	67.15	76.74	2016-08-08 18:10:43
10		davidalves.PhoenixTeam 0.54	65.83	74.42	2016-08-08 18:10:43
11		kid.team.OmegaSquad .0.2	64.26	67.44	2016-08-08 18:10:42
12		myl.micro.TroodonPack 1.10	64.14	72.09	2016-08-08 18:15:31

(b)

Name	mn.CombatTeam 3.23.1	Score Distribution
Flag	Brazil	
APS	83.7483717868	
PWIN	97.6744186047	
ANPP	98.3794627425	
Vote	44.7674418605	
Survival	96.1052199047	
Pairings	43	
Battles	939	
Latest Battle	2016-04-27 09:54:44 UTC	

mn.CombatTeam 3.23.1 Compare

Opponent APS (X) vs. Pairing (Y)

Figura 27 - (a) Ranking TeamRumble (b) Detalhes robô do time brasileiro.

Na competição mundial Robocode, existe o tanque DrussGT, o atual campeão da categoria. Desenvolvido por Skilgannon, DrussGT[22] está no topo da categoria (*AdvancedRobot* – Programação de Robôs Avançados) desde de 2008 e no ano de 2013 ganhou seu estado da arte, com históricos, versões e explicações das técnicas utilizadas no site oficial do Robocode. Seu código é disponibilizado nos termos da *Open Source*, podendo ser estudado, compartilhado e adaptado seguindo os critérios da licença GPL - GNU *General Public License*[23] (emitir o crédito em seu código fonte) (RoboWiki, 2015).

Existem vários torneios Robocode no mundo em países como: Tailândia, Bélgica, Austrália e Nova Zelândia. Dentre disputas ao redor mundo, é possível citar como destaque o torneio da Irlanda *GamesFleadh*[24]. O torneio da Irlanda foi iniciado pela ICS (*Irish Computer Society*), em 2006 denominado "*Robocode Challenge Trophy*" e conta com a participação de estudantes de todo país. Cursos de informática e afins (médio e superior) criam grupos de pesquisa e treinam suas equipes formadas por alunos dos primeiros anos das disciplinas de programação. O torneio da Irlanda proporciona uma emocionante competição com envolvimento das instituições acadêmicas do país. O torneio conta com importantes incentivadores dos eventos ao longo dos anos: Microsoft (atual); EA–*Eletronic Arts* (atual); Dell; Pearson *Education*; Lenovo, etc.

[22] DrussGT. Disponível em: <http://robowiki.net/wiki/RoboRumble>. Robô do Robocode com estado da arte que ocupou o primeiro lugar geral (2008~2013) na categoria "um contra um" no *RoboRumble*.
[23] GNU General Public License (Licença Pública Geral): <https://www.gnu.org/licenses/gpl-howto.en.html>
[24] GamesFleadh. (2016) *Digital Games Programming Festival* - Robocode. Disponível em: <http://gamesfleadh.ie/robocode>.

5.2. Histórico e Competições no Brasil

No Brasil, a instituição Escola Técnica Estadual - Centro Paula Souza (ETEC) possui vários torneios descentralizados em suas unidades espalhadas no estado de São Paulo. Informações da ETEC indicam que existe o estudo para organizar uma liga de Robocode com todas escolas da rede que possuem cursos na área de Informática. Os Institutos Federais também possuem histórico de realizações de disputas de programação com Robocode. Várias universidades (públicas e privadas) espalhadas no Brasil já tiveram iniciativas com o torneio dentro das disciplinas de programação ou em semanas de ciência e tecnologia. A partir de pesquisas de informações registradas em domínios oficiais institucionais, a Tabela 7 exibe algumas instituições brasileiras que realizaram torneios Robocode ao longo dos últimos anos (a Tabela completa pode ser visualizada no Apêndice III). As instituições do topo da tabela representam aquelas com maiores ocorrências de torneios.

Tabela 7 - Ocorrências Torneio Robocode Brasil (Google, 2015).

Instituição	**Cidades**	**Ano(s) do Torneio(s)**
COTIL-UNICAMP - Colégio Técnico de Limeira	Limeira-SP	2015, 2014, 2013, 2012, 2011, 2010
FT-UNICAMP - Faculdade de Tecnologia	Limeira-SP	2015, 2014, 2013, 2012, 2011, 2010
ETEC Centro Paula Souza - João Belarmino	Amparo-SP	2014, 2013, 2012, 2011
IFSP - Instituto Federal São Paulo	Capivari-SP	2015, 2014, 2011
IFAL - Instituto Federal de Alagoas	Arapiraca-AL	2015, 2014
IF Farroupilha	São Borja-RS	2015, 2014
UFU - Universidade Federal de Uberlândia	Patos de Minas-MG	2013, 2012
UFMA - Universidade Federal do Maranhão	São Luís-MA	2014, 2009
UFC - Universidade Federal do Ceará	Quixadá-CE	2011, 2007
ETEC Centro Paula Souza - Armando Pannunzio	Sorocaba-SP	2015
ETEC Centro Paula Souza - Fernando Prestes	Sorocaba-SP	2015
ETEC Centro Paula Souza Adolpho Berezin	Mongaguá-SP	2015
IESP Instituto de Ensino Superior da Paraíba	Cabedelo-PB	2015
IFSP - Instituto Federal São Paulo	Boituva-SP	2015
CETEC Santa Efigênia Centro Paula Souza São Paulo	São Paulo-SP	2014
ETEC Centro Paula Souza - Lauro Gomes	São Bernardo do Campo-SP	2014
ETEC Centro Paula Souza - Prof. José Ignácio Azevedo	Ituverava-SP	2014
ETEC Centro Paula Souza José Luiz Viana Coutinho	Jales-SP	2014
ETEC Centro Paula Souza Prof. Mário Antônio Verza	Palmital-SP	2014
IBTA - Instituto Brasileiro de Tecnologia Avançada -	São Paulo-SP	2014
IFG - Instituto Federal Goiás	Luziânia-GO	2014
IFSC - Instituto Federal Santa Catarina	São José-SC	2014
IFSP - Instituto Federal São Paulo	Campus do Jordão-SP	2014
UNIPAR - Universidade Paranaense	Francisco Beltrão-PR	2014

A classificação da Tabela 7 segue em ordem decrescente de incidências. A pesquisa foi realizada no final do ano de 2015 no site de buscas do Google (2015) em domínios brasileiros com as seguintes expressões: “Torneio Robocode”; “Liga Robocode”; “Campeonato

Robocode" e "Competição Robocode". A Figura 28 exibe algumas divulgações de competições Robocode ao longo dos últimos anos. O Anexo I exibe links dos domínios oficiais das instituições com os resultados registrados acerca da incidência de competições Robocode.

Figura 28 - Divulgações de Algumas Competições Robocode no Brasil.

Notícias veiculadas no site oficial do IFAL - Instituto Federal de Alagoas (Campus Arapiraca) indicaram que mais de 60 alunos, divididos em 16 equipes dos cursos de informática e eletrônica estiveram presentes para realização do primeiro torneio Robocode da instituição. Com esse resultado, o IFAL poderia ser enquadrado como maior torneio em número de participantes e equipes realizado no Brasil até o primeiro semestre de 2015. Os novos números apurados no presente trabalho, indicam que Liga LIAG Robocode 2015 ultrapassou a marca do IFAL em número de participantes, equipes e instituições envolvidas. É possível observar na Figura 29 o mapa de históricos dos torneios Robocode realizados em instituições brasileiras.

Figura 29 - Mapa de Torneios Robocode no Brasil (2015).

5.3. Histórico das Competições Robocode no LIAG

O torneio LIAG Robocode ocorre desde de 2010, em um esforço do Laboratório em promover melhores práticas na área de robótica, inteligência artificial e programação de computadores. O primeiro torneio foi composto por 5 equipes e com envolvimento de aproximadamente 20 alunos.

Os torneios Robocode do LIAG são regidos por regras de modo a estimular a competitividade, administrar as rodadas e garantir a organização. Algumas das principais regras são: (1) Equipes de no máximo quatro participantes; (2) Formada por uma fase classificatória de pontos corridos e outra com enfrentamento direto em quadrangulares (quartas de final, semifinal e final); (3) Ao final de cada rodada, os códigos fontes dos robôs são disponibilizados e divulgados pela organização do evento; (4) Mudanças nos códigos devem ser explicadas para atender as boas práticas de programação; (5) Grupos com códigos copiados são banidos da competição.

A partir de 2012 o LIAG passou a organizar Ligas. O conceito da Liga é promover Torneios descentralizados seguindo o mesmo padrão de regras e organizar uma final da liga entre os melhores colocados dos torneios.

No início do segundo semestre de 2015, o LIAG promoveu três Torneios entre instituições (COTIL-UNICAMP, FT-UNICAMP e IFSP) regidos sob as regras da liga. O LIAG promoveu os torneios na FT-UNICAMP e COTIL-UNICAMP com a condução e supervisão de integrantes e colaboradores do laboratório, sendo o primeiro aberto, contando com a participação de equipes da FT-UNICAMP e externas e o segundo fechado apenas para alunos do Colégio Técnico de Limeira (COTIL). Professores e alunos bolsistas de projetos relacionados ao Robocode do IFSP campus Capivari-SP administram o torneio do IFSP. Todas informações das disputas (resultados, *ranking*, disputas, números, fotos, vídeos etc.) foram centralizadas e disponibilizadas na *web* em um canal oficial do evento (Figura 30). Foram criados calendários distintos para cada um dos torneios locais: COTIL-UNICAMP, de agosto a outubro; FT-UNICAMP, setembro e outubro; IFSP, setembro e outubro com finais realizadas na semana de ciência e tecnologia. Cada torneio intermediário classificou o seu campeão para disputar a final da Liga.

No final do segundo semestre de 2015, o LIAG recebeu as equipes campeãs dos torneios para organização e realização da final da Liga LIAG Robocode. A Liga foi interinstitucional por conta do envolvimento da Faculdade de Tecnologia da UNICAMP (FT-UNICAMP), Colégio Técnico de Limeira (COTIL-UNICAMP) e Instituto Federal de São Paulo (IFSP). A final da liga contou com a participação de equipes vencedoras do torneio do Instituto Federal Farroupilha (IF Farroupilha) do campus de São Borja-RS, disputado junto ao torneio da FT-UNICAMP. Com essa característica interinstitucional os vencedores dos torneios intermediários disputam a liga dos campeões.

Figura 30 - Canal de oficial do evento Liga LIAG Robocode 2015.

A liga e os torneios do Robocode LIAG 2015, tiveram como *slogan* "*Build the best - destroy the rest!*", o mesmo idealizado por Matthew A. Nelson, desenvolvedor do Robocode. No ano de 2015, os eventos contaram com a participação de aproximadamente 30 equipes, totalizando mais de 100 competidores, em três torneios intermediários concomitantes e uma liga interinstitucional. A Tabela 8 indica uma visão geral das equipes por instituições e cidades.

A liga dos campeões foi composta por desafios diretos em nível de semifinal e final com os campeões de cada torneio intermediário. Os segundos melhores colocados de cada instituição tiveram a oportunidade de participar de uma "repescagem" e entrar no quadrangular final em busca do título de campeão. A Figura 31 exibe os campeões dos torneios intermediários e o campeão da repescagem, que formaram as semifinais e a final. A equipe campeã da I Liga LIAG Robocode 2015 foi do IF Farroupilha, do Campus da cidade de São Borja-RS.

Os torneios e a liga podem ser patrocinados e oferecer incentivo em forma de premiações. A Liga LIAG Robocode 2015 contou com incentivos: (1) Destinados aos campões dos torneios intermediários; (2) Para os três primeiros colocados da liga e; (3) Melhor código dentre todas equipes participantes. O laboratório de pesquisa LIAG promoveu a liga no de 2015 e incentivou os campeões intermediários com cursos especializados. A Figura 32 exibe a reunião na FT-UNICAMP para disputa da final da liga LIAG Robocode 2015.

Tabela 8 - Equipes Robocode 2015 por Instituições e Cidades.

Instituição	Equipes Cadastradas	Cidade
FT-UNICAMP	Robot Hunters	Limeira-SP
	Gear	Limeira-SP
	GER	Limeira-SP
	AsimovTeam	Limeira-SP
	JavaCode	Limeira-SP
	Karamba	Limeira-SP
IF Farroupilha	Farrapos	São Borja-RS
	Asad	São Borja-RS
	Infinity	São Borja-RS
COTIL-UNICAMP	UPB-1	Limeira-SP
	The Runaways-6	Limeira-SP
	Baidu-3	Limeira-SP
	Battle Axe-4	Limeira-SP
	Método Destrutor-5	Limeira-SP
	Tortuguitas7	Limeira-SP
	PDeuses	Limeira-SP
IFSP Capivari	Coiotes 404	Capivari-SP
	Ninguém CEGG	Capivari-SP
	SotfDeadCorporation	Capivari-SP
	B&T - Stay Strong	Capivari-SP
	NG Corp.	Capivari-SP
	An Unnamed Team	Capivari-SP
	Allianz Classic	Capivari-SP
	Deathstroke	Capivari-SP
	Overflow	Boituva-SP
	EE-T1 Osório	Hortolândia-SP
IFSP Hortolândia	DeceptiCodes	Hortolândia-SP

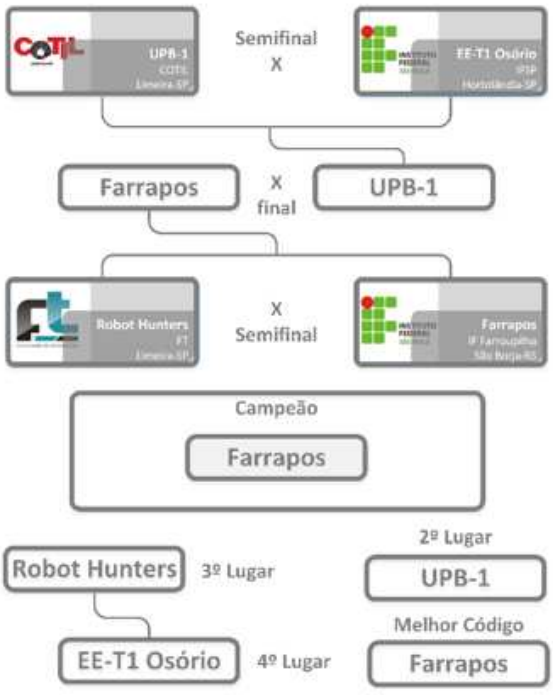


Figura 31 – Resultado Final da Liga Robocode 2015.

Figura 32 – Reunião interinstitucional para disputa final da liga Robocode 2015.

5.4. Robocode Brasil 2016

Robocode Brasil 2016 foi uma competição científica de programação que surgiu da parceria entre a FT-UNICAMP, representada pelo LIAG e o Instituto Federal de Educação, Ciência e Tecnologia (IFSP). O IFSP foi representado pelo campus da cidade de Capivari-SP[25]. A união das instituições FT-UNICAMP e IFSP – Capivari-SP favoreceu a organização do Robocode Brasil.

Entre as metas estabelecidas para Robocode Brasil 2016, apresentou-se como desafios em relação a competição realizada no ano 2015: (1) Dobrar o número de instituições, sendo que em 2015 foram 4 (quatro) instituições participantes; (2) Dobrar número de estados, sendo que em 2015 a liga contou com os estados de São Paulo e Rio Grande do Sul; (3) Dobrar o número

[25] Criado em 1º de fevereiro de 2010, fruto de um termo de compromisso entre: União, representada pelo MEC/SETEC (Ministério da Educação/ Secretaria de Educação Profissional e Tecnológica); Reitoria IFSP; FNDE (Nacional de Desenvolvimento da Educação) e Prefeitura Municipal de Capivari

de torneios, sendo que em 2015, foram 3 (três) torneios concomitantes. A missão da competição relacionou-se em unificar instituições de ensino, de forma a proporcionar alternativas para ensino-aprendizagem de linguagem de programação. A visão do Robocode Brasil 2016 estabeleceu em ser reconhecida como a maior iniciativa de promoção de um evento de competição científica de programação com o Robocode no Brasil.

5.4.1. Gerenciador da Competição Robocode Brasil

Com objetivo de gerenciar informações, foi desenvolvido um sistema *on-line* que representa o canal *web* oficial da competição científica Robocode Brasil 2016 (Lima, 2016). Na página principal do canal existe o logo que identifica a competição, seguida das empresas patrocinadoras e instituições envolvidas na organização do evento. Na mesma página encontram-se informações sobre o ambiente Robocode e as linguagens de programação suportadas na competição (Java e C#). A página apresenta uma breve explanação da organização das competições com *links* para que interessados cadastrem seu e-mail em uma *newsletter* para acompanhar novidades e demais informações. O rodapé da página principal indica a missão, visão, *links* à empresa apoiadora da competição e espaço destinado às redes sociais. As redes sociais funcionaram como extensão de comunicação e divulgação do evento ao público geral.

Para incentivar e disseminar as informações introdutórias e intermediárias do Robocode, o gerenciador dispõe de uma página que abriga um mini curso. Este mini curso foi desenvolvido por um trabalho de conclusão de curso do IFSP no ano de 2015 (Matumoto, Poli, Oliveira, & Borges, 2015). Seu objetivo foi apresentar detalhes dos passos para estudantes em seu primeiro contato com o ambiente Robocode. No curso é oferecida a base necessária para desenvolver os primeiros robôs e entender a mecânica do jogo. Foi planejado e desenvolvido para ser ofertado em tópicos que atendam as características encontradas nas plataformas de ensino a distância. Originalmente hospedado em um Ambiente Virtual de Aprendizagem (AVA), foi remodelado para constar no gerenciador da competição. Não existem pré-requisitos para inscrições no mini curso, que aborda conceitos de lógica de programação, linguagem de programação e programação orientada a objetos, antes de entrar na temática do jogo. De modo intuitivo, estudantes têm acesso aos processos básicos para desenvolvimento dos primeiros projetos na plataforma. Ao final do curso, espera-se que os alunos possam criar seus primeiros protótipos

e que tenham a base necessária para construção de novos projetos. A Figura 33 (a) exibe o gerenciador da competição na plataforma móvel com a identidade visual da competição, enquanto a Figura 33 (b) exibe a página na rede social do Robocode Brasil 2016.

Figura 33 - (a) Gerenciador da Competição (b) Página de Divulgação Rede Social.

Há um *link* para acesso aos "Torneios" no *web* site do canal oficial Robocode Brasil 2016, disponível no menu da página principal, que permite a qualquer pessoa realizar acesso e acompanhar os torneios vigentes concomitantes. É possível observar todas instituições, assim como dados e pontuações das disputas (informações detalhadas dos torneios estão descritas no sub tópico 5.4.3). Ao lado do *link* de torneios está situado o "calendário", com os meses e semanas destinados a realização da competição. A página conta com um "guia de rodadas", desenvolvido para que os participantes tenham acesso às informações simplificadas da organização dos torneios e indicações das características dos níveis de usuários descritos na Tabela 9.

Tabela 9 - Níveis de Usuários do Gerenciador Robocode Brasil.

Nível	Características	Funções no Gerenciador
Gestor do Torneio Local	Colaborador da Instituição (professor, tutor, estagiário, bolsista, etc.). Em geral são professores interessados em inscrever alunos da sua instituição na competição.	(1) Indicar local, dia e horários das disputas da sua instituição; (2) abrir as submissões; (3) encerrar as submissões; (4) publicar resultados das disputas com fotos e vídeos.
Gestor do Torneio Público	Colaborador da Instituição (professor, tutor, estagiário, bolsista, etc.). Um colaborador indicado na equipe do LIAG para gerir torneios interinstitucionais.	(1) Indicar dia e horários das disputas e divulgar link para transmissão *on-line* do evento; (2) abrir as submissões; (3) encerrar as submissões; (4) publicar resultado resultados das disputas com fotos e vídeos.
Líder da equipe	Aluno eleito democraticamente como líder de uma equipe participante de um torneio local ou público.	(1) Acompanhar datas de abertura para submissão (2) submeter os códigos dos robôs da sua equipe (3) acompanhar e conferir os resultados publicados na plataforma.

O link de "*login*" foi exclusivo a alguns participantes devidamente inscritos na competição. Durante o processo de comprovação e aceite das inscrições, são pré-determinados os níveis de acesso conforme atuação na competição (Tabela 9). Para acessar essa área restrita de "*login*" da competição, foram emitidas credenciais sob a forma de usuário e senha e enviadas ao e-mail do participante.

5.4.2. Edital e Regras

Um edital denominado "I Liga Nacional de Robocode" (referente ao Robocode Brasil 2016) foi criado a fim de estabelecer critérios para promoção e organização das disputas entre as instituições de ensino participantes. Uma das atribuições das instituições candidatas à competição foi a ciência das regras estabelecidas no edital, disponibilizado no Apêndice IV. Para realização das inscrições, cada instituição interessada, através de um representante (denominado gestor local), preencheu um cadastro disponível no canal oficial do evento. Existiu um processo de validação de instituições que consistiu em verificar a existência de um gestor local e um mínimo de 4 (quatro) equipes participantes inscritas. A instituição que teve sua inscrição devidamente validada pôde sediar um Torneio Local. O Gestor Local recebeu a função de organização do Torneio Local, assim como ser o elo de comunicação entre instituição e administradores da competição nacional. Quando a instituição não atingiu um mínimo de 4 (quatro) equipes, recebeu um convite para participar do torneio denominado Público. Ao aceitar este convite, a(s) equipes(s) passou(ram) a responder diretamente ao gestor do Torneio Público.

As equipes puderam ser formadas respeitando uma das cláusulas do edital que permite conter de 1 (um) ao máximo de 4 (quatro) integrantes. Professores puderam participar de mais de 1 (uma) equipe como tutores ou orientadores. No ato do cadastro, as equipes selecionaram uma instituição já cadastrada ou solicitaram o cadastro de uma nova instituição quando não a encontraram. Equipes foram vinculadas a um torneio, seja Local ou Público. Para cadastro de equipe na competição, fez-se necessário informar o nome completo e e-mail de cada integrante com a indicação de um Líder (características específicas do papel do Líder são tratadas na subseção de seção 5.4.4.3. Líder de Equipe).

Houve um calendário para competição dividido em duas etapas: (1) Primeira fase denominada de "Torneios", início em 08/08/2016 e término em 19/09/2016; (2) Segunda fase denominada de "Liga Nacional", início em 26/09/2016 e término em 17/10/2016. A fase de "Torneios" foi constituída pelos "Torneios locais" e "Torneio Público". A junção da fase de "Torneios" e "Liga Nacional" formaram a competição Robocode Brasil 2016 (características detalhadas dos Torneios são descritas na sub seção de seção 5.4.3. Torneios). A organização e gestão geral do Robocode Brasil 2016 foi realizada pela equipe do LIAG denominada "Administração da Competição".

As equipes campeãs dos Torneios Locais e a campeã do Torneio Público, receberam vagas para disputar a Liga nacional do Robocode Brasil 2016. Entre os Torneios e a Liga existiu um período estipulado de 1 (uma) semana para que as equipes finalistas pudessem estudar e aprimorar seus conceitos e códigos para disputar a nova fase. Os finalistas, ao retornarem para disputa da Liga nacional, deixaram a condição da gestão institucional e passaram a ser gerenciados diretamente pela equipe de Administração da Competição do Robocode Brasil. Os gestores locais encerraram a atividade junto a sua instituição e receberam um convite para continuar como colaboradores na Liga Nacional. Essa colaboração pôde-se enquadrar no âmbito de acompanhar e/ou representar a sua instituição e equipe, podendo também, havendo interesse, se envolver como mediadores ou juízes para análises e validações de códigos.

A disponibilização do código fonte consistiu um fator predominante em todas edições dos Torneios e Ligas executadas pelos laboratórios LIAG desde 2010. O edital da competição 2016 também estabeleceu que, ao final de cada rodada, o código fonte de cada equipe deve ser disponibilizado aos demais participantes. No Robocode Brasil 2016, essa disponibilização foi efetuada de modo automatizado pelo sistema gerenciador do torneio. O processo ocorre do seguinte modo: (1) Líder da equipe submete o código do robô; (2) Durante o processo de

submissão até a ocorrência da rodada nenhuma equipe tem acesso aos códigos das outras equipes; (3) O sistema é fechado pelo gestor para realização da disputa; (4) Ao término da rodada, resultados e código fonte de todas equipes são disponibilizados. Essa característica contribui para nivelamento e trocas de informações entre equipes. Equipes que possuem códigos com conceitos avançados têm a oportunidade de compartilhar e explicar seus resultados à equipes iniciantes (Meira, Lima, & Borges, 2016).

Existiram ainda regras com especificações técnicas, a fim de delimitar controles específicos ao *framework* de desenvolvimento e ambiente de jogo do Robocode. Essas regras, assim como o plágio de códigos, podem causar desclassificações de equipes. Algumas regras foram herdadas de edições anteriores; outras sofreram modificações e/ou atualizações; novas foram desenvolvidas para edição de 2016 com objetivo de garantir a competitividade entre as equipes e instituições. As principais regras de especificações técnicas podem ser observadas na Tabela 10. Outro fator determinante para definições das regras foi a colaboração das instituições participantes, que trouxeram diferentes aspectos que são adotados mediante consenso.

Tabela 10 - Tabela Regras do Tipo Especificações Técnicas.

Tipo de Regra	Especificações Técnicas
Rodadas	Cada rodada é composta por 3 (três) *rounds* contendo 5 cinco ciclos e intervalos entre cada *round*.
Intervalos	Pausas de 10 (dez) minutos entre *rounds* de cada rodada. É permitido aos integrantes das equipes realizarem alterações em seus códigos ou mesmo substituir o robô.
Tamanho da Arena	Robôs deverão ser inseridos na arena. Rodadas com até 8 (oito) equipes trabalham na proporção de resolução 800 x 800 pixel. Rodadas com mais de 8 (oito) equipes trabalham na proporção de resolução 1200 x 1200 pixel.
Robôs	Robôs podem utilizar apenas classes nativas ao Robocode.

Na Liga LIAG Robocode (realizada em 2015), gestores, colaboradores, equipes, pessoas envolvidas direta e indiretamente no evento foram convidadas a participar da final. Na final houve uma confraternização no laboratório LIAG na instituição sede da liga nacional, FT-UNICAMP, com o propósito de interação interinstitucional e compartilhamento das pesquisas e experiências vividas ao longo da competição. A equipe de Administração da Competição do ano de 2015, do laboratório de pesquisa LIAG, ficou responsável por recepcionar os participantes e promover a disputa final para registrar um novo campeão brasileiro. O evento final contou com a participação dos apoiadores e patrocinadores. Certificados de participação

foram emitidos a todas equipes e certificados de campeões foram emitidos aos primeiros colocados dos torneios locais (1º colocado) e liga nacional (1º, 2º e 3º colocados).

5.4.3. Torneios

Conforme edital e regras, existe a fase de "Torneios", constituída pelos "Torneios Locais" e o "Torneio Público". Tecnicamente, as modalidades das disputas foram iguais, com exatamente a mesma quantidade de rodadas classificatórias e quadrangulares finais, ocorrendo diferenciação apenas na forma de gestão e local de incidência. O Torneio Local foi realizado dentro de uma instituição (com equipes da instituição) e gerido pelo Gestor Local enquanto o Torneio Público foi transmitido *on-line* na web (com equipes de instituições diferentes) e gerido por um Gestor Público integrante da Administração da Competição Robocode Brasil 2016. A Figura 34 mostra exibição dos Torneios no gerenciador da competição.

Figura 34 - Disposição dos Torneios no Gerenciador.

A Figura 35 exibe o calendário dos Torneios com a disposição das rodadas em semanas. Coube ao Gestor Local ou Público determinar um dia da semana, horário e local (caso público um endereço *on-line* de transmissão) para realização de cada rodada. O Torneio foi divido em duas etapas distintas: classificatórias e eliminatórias. A primeira etapa, que ocorreu durante os meses de agosto e início de setembro de 2016, foi realizada com rodadas classificatórias. Estas

rodadas foram dispostas de primeira a quinta rodada, com algumas regras específicas observadas na Tabela 11. A pontuação atribuída a cada rodada disputada durante a etapa classificatória pode ser observada na Tabela 12.

AGOSTO DE 2015						
Dom	Seg	Ter	Qua	Qui	Sex	Sab
	1	2	3	4	5	6
	INSCRIÇÕES					
7	8	9	10	11	12	13
	SEMANA DA PRIMEIRA RODADA					
14	15	16	17	18	19	20
	SEMANA DA SEGUNDA RODADA					
21	22	23	24	25	26	27
	SEMANA DA TERCEIRA RODADA					
28	29	30	31	1	2	3
	SEMANA DA QUARTA RODADA					

SETEMBRO DE 2015						
Dom	Seg	Ter	Qua	Qui	Sex	Sab
				1	2	3
	SEMANA DA QUARTA RODADA					
4	5	6	7	8	9	10
	SEMANA QUINTA RODADA					
11	12	13	14	15	16	17
	SEMANA - SEMIFINAL					
18	19	20	21	22	23	24
	SEMANA -FINAL					
25	26	27	28	29	30	1
	PAUSA					

Figura 35 - Calendário da Fase de Torneios.

Tabela 11 - Regras Específicas a Primeira Etapa Classificatória dos Torneios.

Tipo de Regra	Especificações das Regras
Disputas	Equipes se enfrentam (todos contra todos) dentro da arena do Robocode.
Rounds	Herdado das regras gerais, rodada será composta por 3 (três) *rounds*, com intervalos de 10 (dez) minutos entre cada *round*. A cada intervalo as equipes poderão alterar o código fonte dos robôs, ou ainda substituir os robôs.
Ranking	Ao final de cada *round,* o programa do Robocode determina um *ranking* de colocações que definem as pontuações da rodada.

Tabela 12 - Pontuação Etapa Classificatória.

Colocação	1º	2º	3º	4º	5º	6º	7º	8º	Demais
Pontuação	10	8	6	5	4	3	2	1	0

As equipes que não enviaram códigos ou enviaram fora do prazo receberam a pontuação 0 (zero). Ao término de cada rodada, o Gestor Local ou Público cadastrou a pontuação de cada equipe no gerenciador do Torneio. Foi recomendado, aos Gestores dos Torneios, o registro de vídeos ou fotos ao final de cada rodada. Estas informações, quando fornecidas, foram compartilhadas no canal oficial. Ao final da fase classificatória foram somadas as pontuações de todas as rodadas: as 4 (quatro) primeiras equipes com maiores pontuações passaram para a segunda fase do Torneio.

A segunda etapa dos Torneios, ocorrida nas últimas semanas do mês de setembro de 2016, foi realizada com rodadas eliminatórias, com regras específicas exibidas na Tabela 13. Esta etapa foi constituída por enfrentamentos diretos (um contra um). O primeiro colocado competiu com o quarto colocado (denominado de primeira disputa) e o segundo colocado competiu com terceiro colocado (denominado de segunda disputa). Na última rodada, os respectivos vencedores da primeira e segunda, disputaram a final para determinar o segundo colocado e o campeão do Torneio. Enquanto isso, perdedores da primeira e segunda disputaram a terceira colocação. Campeão de cada Torneio Local e Torneio Público conquistou uma vaga na Liga Nacional. Quando houve mais de 8 (oito) equipes na mesma Instituição (participantes em todas rodadas), classificaram-se para Liga Nacional o primeiro e segundo colocados respectivamente. Em cada torneio existiu a cerimônia de premiação com entrega de medalhas (ouro, prata e bronze) para integrantes das 3 (três) equipes primeiras colocadas.

Tabela 13 - Regras Específicas a Segunda Etapa Eliminatórias dos Torneios.

Tipo de Regra	Especificações das Regras
Disputas	Um contra um.
Rounds	Herdado das regras gerais, rodada será composta por 3 (três) *rounds*, com intervalos de 10 (dez) minutos entre cada *round*. A cada intervalo as equipes poderão alterar o código fonte dos robôs, ou ainda substituir os robôs.
Resultado	Exibido diretamente no programa do Robocode (não existe a tabela de pontuação). Equipe que vencer 2 (dois) *rounds* passa para próxima rodada.

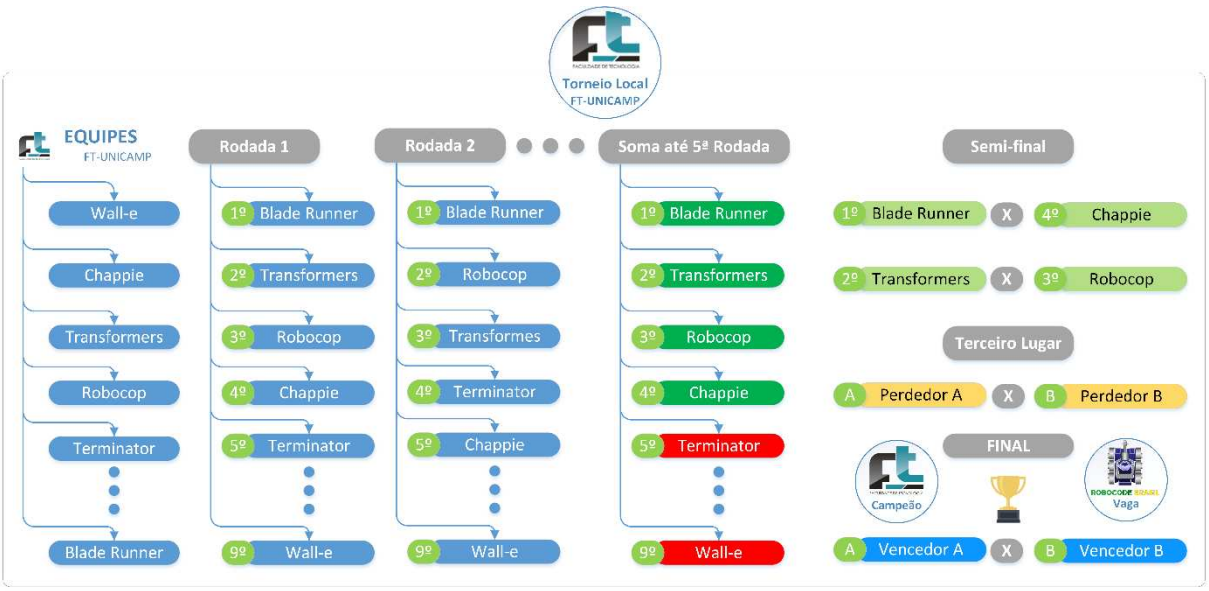


Figura 36 - Visão Simplificada dos Torneios Locais.

A Figura 36 apresenta de modo simplificado a ocorrência de um determinado Torneio. Observa-se etapas presentes nos Torneios: (1) Cadastro das equipes; (2) Etapa de rodadas; (3) Classificação das 4 (quatro) melhores equipes; (4) Etapa eliminatória constituída por semifinais, disputa terceiro lugar e final; (5) Premiação para os 3 (três) primeiros colocados; (6) Campeão do torneio ganha a vaga para disputar a Liga Nacional.

5.4.4. Gestores dos Torneios

Conforme indicado no item do gerenciador da competição (subseção de seção 5.4.1. Gerenciador da Competição Robocode Brasil), existiram determinados participantes das instituições que receberam funções ou níveis específicos perante o evento. Estas funções foram definidas no ato de validação das inscrições e determinação de um mínimo de equipes por instituição. Cada uma das funções foi descrita nas próximas subseções de seção.

5.4.4.1. Gestor do Torneio Local

Gestores dos Torneios Locais foram colaboradores presentes e atuantes nas instituições. Em geral, foram caracterizados por professores que ministram disciplinas nas áreas de informática e afins. Caracterizada por conceitos relacionados às ementas das seguintes disciplinas: lógica de programação; linguagem de programação; programação orientada a objetos; robótica entre outras. Antes mesmo da inscrição, existe um envolvimento da Administração da Competição e Gestor Local com objetivo de formar equipes interessadas na participação do Torneio. Algumas das decisões tomadas para concretização do edital contou com a colaboração e envolvimento dos Gestores Locais com sugestões em entrevistas, conversas, e-mails e videoconferências.

Os Gestores Locais tiveram funções chaves ligadas ao estreitamento da comunicação com a equipe de administração e organização da competição nacional Robocode. Ao final do processo de validação da inscrição de uma instituição, o Gestor Local recebeu a reponsabilidade de garantir as condições necessárias para organização e ocorrência do evento. Para auxiliar a colaboração do Gestor Local, a equipe de Administração da Competição, através do gerenciador da competição, disponibilizou credenciais (usuário e senha) necessárias para acesso a área de administração. De posse das credenciais, o Gestor Local teve acesso a área administrativa da sua instituição. Nesta área, foi possível criar e gerenciar de modo intuitivo e automatizado o

Torneio Local. A área administrativa foi desenvolvida para suprir uma deficiência encontrada na competição do ano de 2015, na qual os torneios eram criados e gerenciados de forma manual, com dispêndio de tempo excessivo para esse fim.

Para o presente trabalho, Gestores Locais colaboraram como elo entre apresentar os desafios inerentes da competição que se enquadrem na PBL e aferir as questões que puderam caracterizar indícios motivadores entres os participantes. Para apresentar estes resultados, este trabalho sugeriu a emissão de questionários aos alunos participantes que foram disseminados e acompanhados pelos Gestores Locais. Os gestores tiveram importantes contribuições ao responderem questões específicas das situações pertinentes aos torneios (presentes no Capítulo 6. Resultados e Discussões).

Em funções de organização, para determinação de um Torneio Local, o gestor indicou um mínimo de 4 equipes locais participantes. Coube ao gestor analisar o calendário das semanas de rodadas para definição dos dias e horários das disputas. Recepcionou localmente as equipes nos dias e horários definidos para execução das rodadas locais até a final do torneio. Foi função do Gestor o acesso ao sistema gerenciador da competição com suas credenciais a fim de informar: (1) Dia, local e horário da ocorrência dos Torneios; (2) Abertura do sistema para submissão dos códigos; (3) Encerramento do prazo de submissão; (4) Publicação dos resultados assim como o envio de fotos e/ou vídeos das rodadas. A Tabela 14 exibe detalhes das atividades do Gestor Local perante a área administrativa do sistema gerenciador da competição. Essas atividades receberam o nome de *status*, conforme evolução dos processos e execução das rodadas no gerenciador. A Figura 37 indica a tela de *status* exclusivo da administração do Gestor do Torneio Local com indicações a sua instituição.

Tabela 14 - Detalhamento das Atividades em Relação ao Status.

Status	**Detalhamento Atividades**
Futura	Caracterizada por todas rodadas que ainda não ocorreram.
Informativa	São informados datas, local e horários da partidas (no caso do torneios públicos ou não presenciais, indicativa do link *on-lin*e canal do transmissão ao vivo, ex. Youtube; Skype, etc.).
Submissão	O gestor (local ou público) determina a data de início das submissões dos códigos das equipes.
Fechada	Não será mais possível que os líderes das equipes submetam seus códigos.
Publica	Gestores devem indicar: resultado das rodadas; descrever informações pertinentes a rodada; imagens e vídeos das rodadas. Obs: ao clicar em publicar, todos os dados informados serão divulgados no site do Robocode Brasil.

Figura 37 - Ambiente do Administrador com Funções da Gestão Local.

5.4.4.2. Gestor do Torneio Público

Consistiu em função similar ao Gestor Local, que atendeu as mesmas características perante o gerenciador da competição no que diz respeito ao *status*. Sua diferenciação ocorreu quando instituições não conseguiram indicar um mínimo de 4 (quatro) equipes para formar um torneio local, porém houve interesse em participar da competição. As instituições que possuíam de 1 a 3 equipes interessadas em participar do evento, indicaram suas equipes ao Gestor do Torneio Público. Neste cenário não existiu figura de um Gestor Local, e as equipes inscritas ficaram subordinadas ao Gestor do Torneio Público. Portanto, o Gestor do Torneio Público ficou condicionado a funções de comunicações específicas no contato direto com várias equipes, de diferentes instituições, devidamente validadas no ato de inscrição.

5.4.4.3. Líder de Equipe

Cada equipe inscrita no torneio Local ou Público elegeu democraticamente um Líder. Esse representante foi um *stakeholder* fundamental na competição. Competiu ao Líder, a intermediação da comunicação entre a equipe e os Gestores de suas respectivas competições. Cada líder de equipe recebeu uma credencial de acesso ao gerenciador da competição. Através do acesso, Líder recebeu responsabilidades: (1) Acompanhar a abertura de cada rodada do torneio; (2) Verificar data, local e horário de ocorrência da competição; (3) Submeter código da equipe nas aberturas de cada rodada.

O Líder desenvolveu um papel fundamental ao longo da ocorrência da competição. Podendo fomentar um ambiente na equipe propício à interação com propósitos: (1) Gerar debates de acordo com desafios encontrados e resultados conquistados no jogo e na competição; (2) Determinar, juntamente com a equipe, os objetivos e estratégia de jogo (ex. melhoria nos métodos de movimentação do robô ou acurácia no disparo); (3) Discutir pontos fortes e fracos ao final de cada rodada disputada; (4) Incentivar pesquisas por situações favoráveis para serem utilizadas no jogo (ex. estudar conceitos ou métodos de programação ainda inexplorados). Líderes predominantemente ativos em estimular ações nas equipes têm maiores possibilidades de sucesso durante as disputas ao longo da competição.

5.4.5. Liga Nacional

Equipes campeãs dos Torneios Locais integraram as instituições participantes da Liga Nacional. A Liga permitiu o encontro interinstitucional de equipes do Brasil. Para disputa entre instituições, foram criados 2 (dois) grupos através de um sorteio (Figura 38). Na Liga, houve a disputa similar a etapa classificatória dos "Torneios", com enfrentamentos "Todos Contra Todos" para definir um ranking dos 4 (quatro) melhores colocados de cada grupo. Mesmo com a existência de mais de 4 (quatro) equipes por grupo, foram classificadas apenas os 4 (quatro) melhores resultados.

No final da Liga, ocorreu uma cerimônia de premiação com entrega de medalhas (ouro, prata e bronze) para os 3 (três) primeiros colocados. Foram entregues também aos 3 (três) primeiros colocados premiações em forma de brindes oferecidos pelos patrocinadores.

Figura 38 - Visão Macro da Liga Nacional.

6. RESULTADOS E DISCUSSÕES

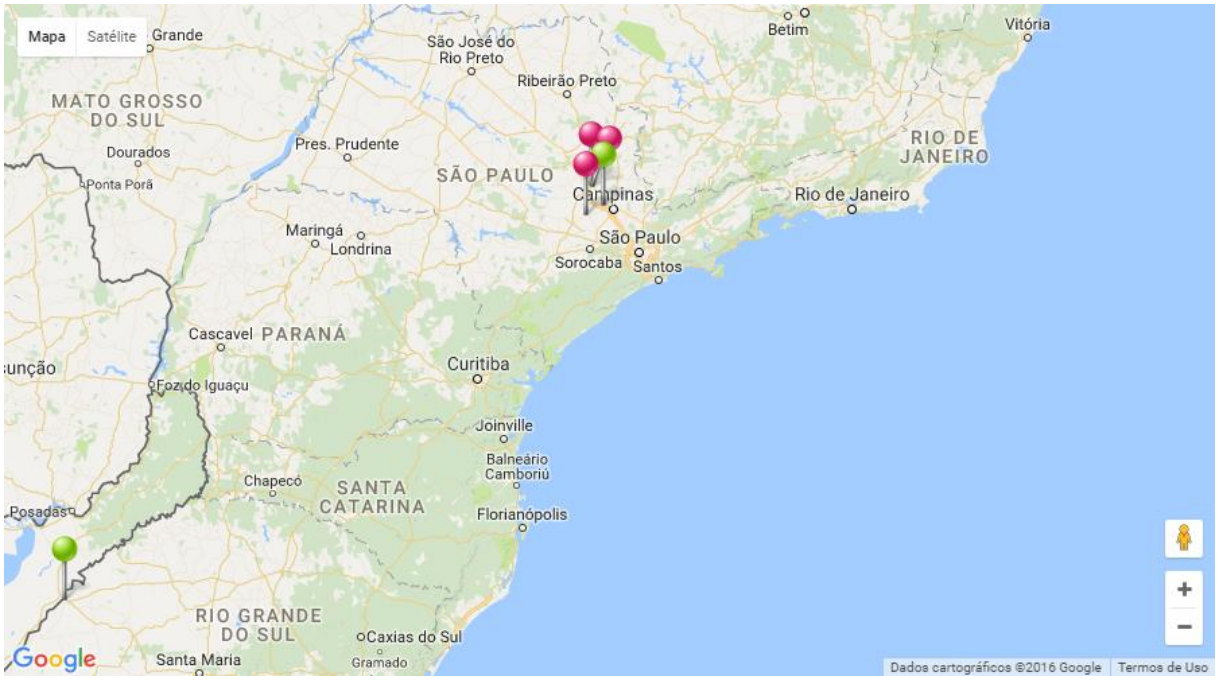
Este capítulo apresenta informações sobre a Liga LIAG Robocode 2015 e sua evolução, o Robocode Brasil 2016, comparando os resultados. Mapas de referência das localizações das instituições participantes dos Torneios Locais e Público no Brasil são apresentados. São apresentados os resultados das entrevistas, como base de dados, das experiências dos professores participantes dos Torneios. É apresentada a pesquisa-ação, com os resultados obtidos a partir do levantamento e discussão das incidências de falhas/problemas de gestão da edição anterior da competição (Liga LIAG 2015). A partir das discussões dos problemas detectados em 2015, ainda com base no método da pesquisa-ação, são apresentadas soluções para desenvolvimento e gestão da competição de 2016 (Robocode Brasil 2016).

É descrito a metodologia do PBL e sua utilização na competição com definições de funções específicas de colaboração entre gestores e alunos. Apresenta os Gestores dos Torneios com funções associadas em indicar aos alunos possibilidades de implementação de modelos de estruturas para determinação dos problemas gerados no jogo. Há a caracterização do PBL na competição e seu emprego na análise e delimitação dos problemas de jogo relacionados aos conceitos introdutórios da linguagem de programação. É exibido a caracterização do problema, da qual alunos com auxílio dos gestores contextualizam o desenvolvimento dos conceitos e práticas com base no método LBD.

São apresentadas as técnicas de pesquisa de campo e observação direta, utilizadas para verificar as motivações na participação da competição. Apresenta a ocorrência da pesquisa de campo a partir da visita em instituições para coleta de dados no momento da ocorrência da competição Robocode Brasil 2016. O presente capítulo apresenta a observação direta extensiva com a aplicação de 2 (dois) questionários aos alunos participantes dos Torneios do Robocode Brasil 2016. O primeiro, denominado de Questionário Inicial, aplicado antes e durante as primeiras rodadas dos Torneios. Neste questionário, são apresentadas as questões pertinentes à disciplina de linguagem de programação e motivações relacionadas à competição. O segundo, denominado Questionário Final, esteve disponível nas últimas rodadas e ao término dos Torneios. O segundo questionário apresenta questões com relações entre as impressões e resultados conquistados com as experiências e participações durante a competição. No presente capítulo adotou-se a prática de equivalência de significados para os termos: problemas e desafios.

6.1. Comparativo entre torneios

Ocorreram comparativos entre as competições Liga LIAG Robocode 2015 e Robocode Brasil 2016. A Figura 39 exibe um mapa com informações referenciadas da competição interinstitucional Liga LIAG Robocode 2015 (detalhes da competição de 2015 foram tratados na subseção 5.3. Histórico das Competições Robocode no LIAG). Em 2015, o quadro geral da competição apresentou-se: total de 5 (cinco) instituições participantes; cerca de 100 (cem) alunos; 30 (trinta) equipes; 3 (três) Torneios Locais (representados na Figura 39 na cor rosa) e 2 (duas) instituições com equipes no Torneio Público associado à FT-UNICAMP (representados na Figura 39 na cor verde). No ano de 2015 foram alcançados 2 (dois) estados (São Paulo e Rio Grande do Sul). Firmou-se um contrato de patrocínio antes do início das competições. O patrocínio, do ano de 2015, contemplou premiações sob a forma de brindes e cursos de programação nas plataformas Android[26] e iOS[27]. O fator premiação foi apresentado como um dos fatores de motivação dos alunos participantes na competição de 2015 (Meira et al., 2016).



[26] https://www.android.com. Android consiste um Sistema Operacional baseado em Linux originalmente desenvolvido para atender dispositivos móveis.

[27] http://www.apple.com/ios. iOS consiste um Sistema Operacional para dispositivos móveis da empresa Apple® Inc.

Figura 39 - Mapa referenciado das Instituições da Liga LIAG de 2015

Em comparação ao ano de 2015, a Figura 40 exibe o Rocode Brasil 2016. Observa-se no mapa o aumento na quantidade de instituições, passando de 5 (cinco) para um total de 14 (quatorze) instituições participantes, envolvendo cerca de 170 (cento e setenta) alunos inscritos e 50 (cinquenta) equipes. A porcentagem de inscrições por instituições pode ser visualizada na Figura 41. O Robocode Brasil 2016 consistiu na maior competição brasileira com ambiente Robocode, ultrapassando sua antecessora (Liga LIAG Robocode 2015). Apresentou 7 (sete) Torneios Locais (instituições na cor rosa na Figura 40) e 1 (um) Torneio Público (instituições na cor verde na Figura 40). Comparando-se o mapa da Figura 39 (2015) com mapa da Figura 40 (2016), observa-se o crescimento da competição. Houve também um aumento no número de estados participantes, de 2 (dois) para 6 (seis) estados, incluindo: (1) Rio Grande do Sul; (2) Santa Catarina; (3) Paraná; (4) São Paulo; (5) Minas Gerais; (6) Bahia.

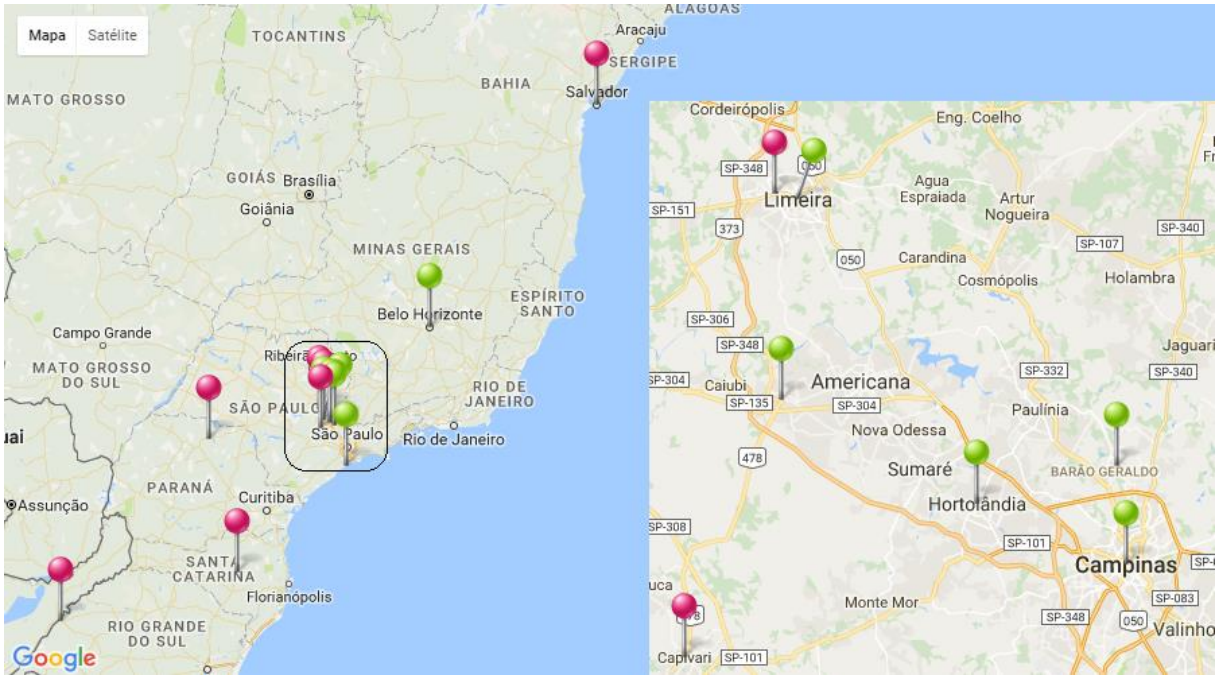


Figura 40 - Mapa Referenciado da Instituições do Robocode Brasil 2016.

A Figura 42 exibe a disposição das instituições nos torneios locais e público, com a identidade visual da instituição e cidade. Uma lista completa com as equipes do Robocode Brasil 2016 classificadas por instituições, cidades e tipo de Torneio pode ser visualizada no Apêndice V. Em contrapartida ao ano de 2015, o patrocínio do Robocode Brasil 2016 foi firmado após o encerramento das inscrições, quando os torneios encontravam-se em andamento. Portanto, a divulgação do Robocode Brasil 2016 ocorreu sem o atrativo das

premiações sob a forma de brindes ou cursos. Com advento da conquista tardia de um patrocinador Ouro[28] para evento, as instituições e equipes aderiram as inscrições com outras motivações que não contemplaram premiações.

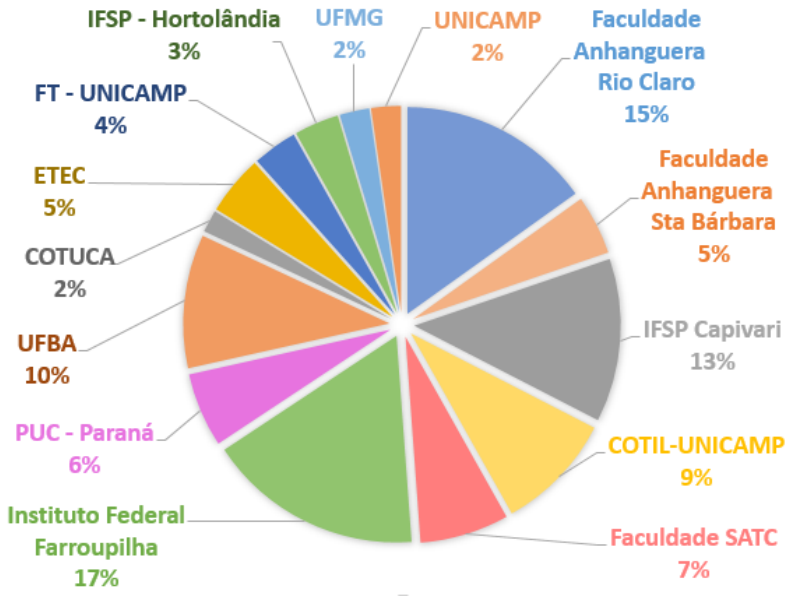


Figura 41 - Porcentagens de Inscrições no Robocode Brasil 2016

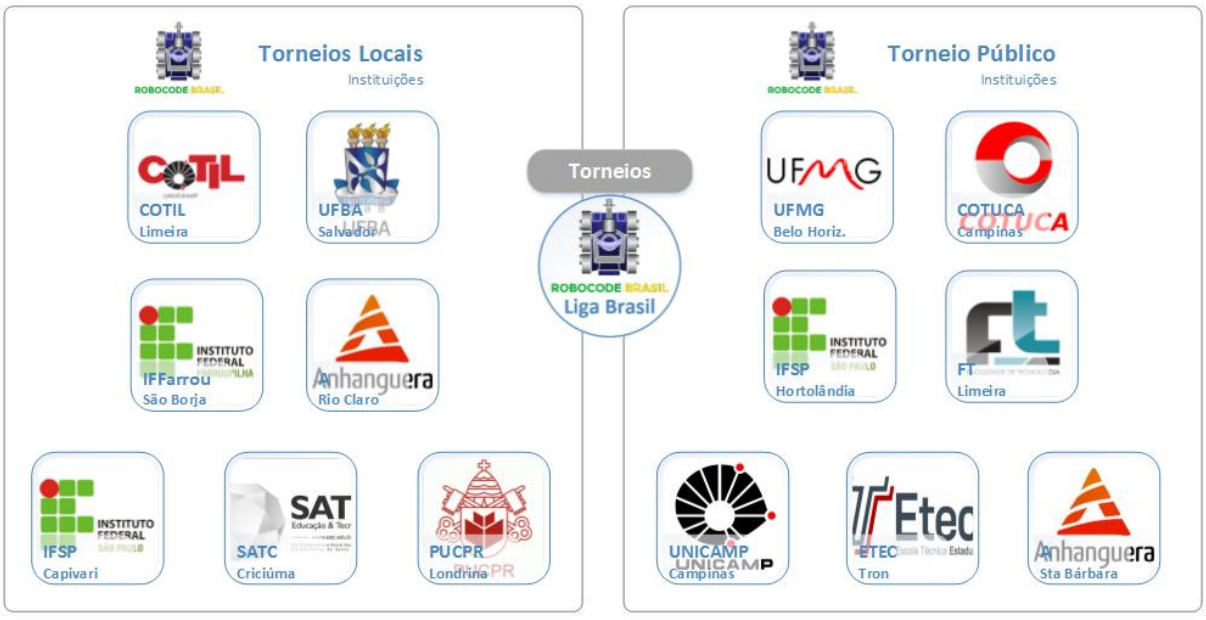


Figura 42 - Instituições Pertencentes aos Torneios Locais e Público.

[28] Empresas patrocinadores de eventos, em geral, têm modelos de patrocínios: Ouro; Prata e Bronze conforme cotas de apoio. No caso do Robocode Brasil 2016 existiu uma empresa Ouro patrocinadora.

6.2. Características do PBL na competição

A presente subseção associa as metodologias dos conceitos apresentados no referencial teórico (Capítulo 3) como a PBL e a GBL aplicados nos jogos da competição Robocode Brasil 2016.

Em geral, professores recebem a ementa da disciplina, exemplo da linguagem de programação, e têm a função de elaborar plano de ensino e aulas a partir do conteúdo estabelecido no plano de referência curso. O conteúdo deve chegar aos alunos respeitando determinados limites de tempo durante a aplicação da disciplina. Os tempos para tratar conteúdos nas disciplinas de linguagem de programação podem variar e são geralmente estipulados com base na quantidade de aulas e complexidade dos conceitos. Com definições de quantidade de tempo, disponibilizada para tratar cada conteúdo, professores têm a missão de eleger métodos adequados de ensino-aprendizagem, que normalmente remetem as aulas tradicionais clássicas e práticas de conteúdos em laboratórios. Porém, antes mesmo de entrar em métodos de ensino-aprendizagem, é importante ressaltar que cada aluno possui seu próprio tempo de aprendizagem. Respeitando o ritmo de cada aluno, é improvável trabalhar um mesmo conteúdo, da mesma forma e com a mesma fatia de tempo para uma turma de alunos e ao final exigir uma única resposta (Becker, 1993).

O presente trabalho considera que os processos de ensino-aprendizagem são complementares e que podem ser enriquecidos por métodos que apresentem características pouco encontradas no ensino tradicional. Neste contexto, este trabalho inseriu processos auxiliares a fim de estimular a aprendizagem de conceitos de linguagem de programação com jogo educacional digital em uma competição científica. O Robocode Brasil permitiu a utilização de metodologias como a PBL ou GBL respectivamente. As condições desenvolvidas dentro do jogo como criação dos robôs e as características presentes nas regras da competição geraram, naturalmente, desafios durante as disputas. Os desafios gerados caracterizaram os problemas tratados nas metodologias.

As características importadas da metodologia GBL referem-se na utilização de jogos digitais em contextos educacionais com objetivo de auxiliar os processos de aprendizagem, com a motivação dos alunos. O contexto educacional do presente trabalho, remeteu à competição científica de programação com uso do jogo digital Robocode.

6.2.1. Atuação dos Gestores e Líderes com a PBL

Durante a elaboração das regras da competição, ficou estipulado que cada Instituição que tivesse um mínimo de 4 (quatro) equipes inscritas teria o direito a sediar um torneio local. A validação da Instituição que atendeu esses pré-requisitos consistiu em troca de informações entre o gestor local e equipe de Administração da Competição Robocode Brasil. Foram utilizados variados canais de comunicação para intermediar essa troca de informações: e-mails; videoconferências; canais nas redes sociais; telefones etc. Os gestores indicaram os canais disponíveis mais favoráveis para a conversa e administradores entraram em contato para estabelecer comunicação. Independentemente do canal escolhido, as mesmas informações foram tratadas com cada um dos Gestores responsáveis por Torneios Locais. A Tabela 15 indica o fluxo de informações do Gestor Local para a Administração da Competição com propósitos na identificação institucional e procedimentos realizados no gerenciador da competição. Para o Torneio Público, quando não atingido mínimo de equipes por instituição, foi usado o mesmo fluxo de informações da Tabela 15, porém por intermédio do Líder da equipe ao Gestor do Torneio Público.

Tabela 15 - Informações da Instituição e Gestor Local.

Institucional	Resultado da Informação
Tipo de Instituição	Pública / Privada.
Informações Institucionais	Nome e local da instituição
Identificação do Gestor	Nome, contato e função na instituição (ex. professor de disciplinas de linguagem de programação).
Cursos Oferecidos	Informações sobre oferta de cursos na área de informática
Motivações	Quais características levou a instituição a participar da competição.
Alunos Participantes	Informação sobre alunos inscritos e a quais cursos estão vinculados.
Definições das equipes	Como ocorreu o método de definição das equipes.

Informações foram classificadas e transmitidas com objetivos principais e específicos, entre a Administração da Competição e os Gestores Locais, e entre o Gestor Público e os Líderes de equipes. As informações com objetivos principais centraram-se em temas: acompanhar disputas de acordo com o calendário estipulado no edital; descobrir as motivações das equipes ao longo das disputas. As informações com objetivos específicos foram relacionadas a apresentação e aplicação dos desafios (problemas) a cada rodada da competição. No caso da competição Robocode Brasil, o papel dos professores, ou tutores, foram

representados pelos Gestores com a missão de auxiliar nos apontamentos dos problemas gerados durante o jogo.

Um Gestor possui como papel fundamental proporcionar um ambiente adequado para que as equipes possam identificar os problemas, discutir hipóteses de soluções e trabalhar de modo ativo na percepção das questões envolvidas. As situações-problemas geradas na execução dos Torneios, durante os jogos, e a atuação dos Gestores como professores com objetivo de proporcionar acesso aos conceitos necessários para solução configuram uma das características marcantes da PBL. Em geral, os Gestores foram professores das disciplinas da área de programação e afins (tratado na subseção de seção 5.4.4. Gestores dos Torneios). O seguinte processo de informação foi transmitido aos Gestores: (1) Importância em acompanhar disputas; (2) Promover ambiente lúdicos; (3) Propiciar, aos alunos, condições para solucionar problemas do jogo. Ao final do processo de transmissão, os Gestores receberam a capacitação adequada para operar o gerenciador da competição (apresentado na subseção de seção 5.4.1. Gerenciador da Competição Robocode Brasil). A Tabela 16 indica o fundamental papel do Gestor na competição e características. A delimitação e definição dos problemas específicos presentes na disciplina de linguagem de programação são tratados nas próximas subseções.

Tabela 16 - Funções e Características do Gestores Locais.

Função do Gestor	Características da Função
Divulgar a competição	Apresentar o modelo da competição Robocode Brasil na Instituição.
Definir ambiente	Criar um ambiente adequado para ocorrência da competição. Em geral, as competições são disputadas em local com infraestrutura adequada como laboratórios de informática.
Conscientizar e acompanhar a preparação das equipes	Apresentar e edital da competição para que as equipes possam definir estratégias de disputas com base nas regras. Acompanhar a preparação de cada equipe antes mesmo do início da competição. Esse processo permite que novos conceitos de linguagem de programação sejam previamente apresentados.
Criar as rodadas	Marcar as datas de cada rodada e incentivar os alunos a desenvolver e postar seus robôs no gerenciador da competição.
Acompanhar as rodadas	Regras permitem um determinado tempo para analisar os problemas encontrados entre as disputas dentro de uma rodada. Neste intervalo de tempo o robô pode ser substituído ou reprogramado se a equipe julgar necessário.
Acompanhar as equipes antes, durante e após cada rodadas	Nos intervalos de tempo cada equipe terá demandas de problemas diferenciados no jogo (ex. equipes vencedoras de *rounds* devem manter a sequência; equipes que não apresentam resultados satisfatórios devem rever a programação de seus robôs). Neste momento entra o papel do gestor afim de intermediar debates de possíveis soluções que podem resultar na apresentação conceitos de programação.

6.2.2. Delimitação dos Problemas

É sabido, de acordo com as regras contidas no edital da competição 2016 (tratado na subseção de seção 5.4.2. Edital e Regras), que ao final de cada rodada, os códigos das equipes foram disponibilizados no gerenciador da competição. Essa característica pôde ser amplamente explorada pelo Gestor de forma a incentivar a análise de códigos de outras equipes da instituição ou mesmo equipes de outros Torneios. Essas análises puderam auxiliar equipes a compreenderem como outras equipes abordaram os conceitos de linguagem de programação.

Com o objetivo de analisar assntos introdutórios, o presente trabalho foi delimitado nos seguintes conceitos de linguagem de programação Java: (1) Sintaxe; (2) variáveis; (3) Estruturas condicionais; (4) Estruturas de repetição. A competição permitiu a utilização de conceitos avançados e deixou os alunos livres para que buscassem soluções de programação de acordo com sua evolução e entendimento dos conceitos. Essa delimitação do ambiente da pesquisa, permitiu elaborar desafios comuns aos alunos iniciantes e àqueles com prévios conhecimentos dos conceitos de programação. Essas definições permitiram, também, a percepção da evolução das equipes no ambiente de desenvolvimento Robocode durante a programação dos robôs com a utilização dos conceitos introdutórios. Com o entendimento dos 4 (quatro) conceitos introdutórios, equipes tenderam a buscar por novos conceitos tendo como objetivo o aperfeiçoamento dos robôs. Gestores foram aconselhados a estarem atentos no momento de maturação da aprendizagem dentro das equipes (com relação ao aprendizado dos conceitos introdutórios) para propor e motivar novos desafios aos competidores.

Alunos foram orientados pelo edital e Gestores a desenvolver e trabalhar seus códigos com base nas melhores práticas de programação. Uma das categorias da competição (Liga) premiou o melhor código. Os comentários explicativos dos passos adotados no desenvolvimento dos códigos (preferencialmente dos conceitos delimitados) e identação dos códigos foram características levadas em consideração na eleição do melhor código. Mesmo os comentários não sendo partes compiladas no programa do robô, puderam promover o entendimento dentre alunos. Os comentários puderam garantir que um integrante realizasse e documentasse suas alterações à equipe. Nos torneios dos anos de 2015 e 2016, essa característica permitiu, que mesmo na falta de integrantes, equipes pudessem realizar novas alterações com as explicações descritas nos códigos.

Boas práticas também indicaram a realização da identação nos códigos dos robôs. Por exemplo, em um programa Java, deve existir o ponto e vírgula como forma de delimitar o final de uma instrução, antes do início de uma nova. É possível, em Java, escrever várias instruções em uma mesma linha, porém é recomendável que cada nova instrução fique em uma linha, para organização do programa. A identação refere-se a organização física das linhas de um bloco de instruções no programa. Essa organização torna-se uma característica fundamental em um ambiente PBL, pois auxilia o entendimento dos programas desenvolvidos na equipe. Isso possibilitou que qualquer um dos integrantes de uma mesma equipe, após definir a forma de identação, estará familiarizado com a organização do programa do seu robô. A Figura 43 exibe parte de um programa com identação e comentários, premiado como melhor código da competição Liga LIAG Robocode 2015. É possível observar na Figura 43 a identação adotada na tabulação da condicional *if* (se) juntamente com as chaves.

```
71      /**
72       * <p>
73       * Adiciona os eventos personalizados ao robo
74       * </p>
75       */
76      private void addCustomMethods() {
77
78      // Verifica se adversário disparou a partir da analise
79      addCustomEvent(new Condition("atirou") {
80
81          public boolean test() {
82          if (!enemy.isAlive()) {
83              double energyDiff = enemy.energyOld - enemy.energy;
84              return energyDiff >= .1 && energyDiff <= 3;
85          }
86          return false;
87          }
88      });
```

Figura 43 - Código Endentado e Comentado.

Além dos comentários e identação presentes nos códigos, outra característica marcante na competição foi a comparação em relação a evolução entre o primeiro código postado e sua comparação com o último código postado. Os gestores foram aconselhados a acompanhar a evolução de todas equipes durante os Torneios. Na fase da Liga Nacional, Gestores Locais e administradores da competição acompanharam a evolução das equipes.

6.2.3. Definição dos Problemas

Os problemas foram gerados automaticamente durante as disputas no jogo Robocode. Variados problemas puderam ser identificados ao final de cada *round* dentro de uma mesma rodada: (1) Movimentação do robô; (2) Robô colidindo na parece; (3) Robô colidindo com outro robô; (4) Acurácia nos disparos; (5) Radar não identifica adversários; (6) Ajustes no radar; (7) Controle nos disparos para poupar energia (8); Coleta inadequada de informações; (9) Robô sofrendo danos excessivos; (10) Ausência de controle de energia, entre outros. Assim, como o presente trabalho foi delimitado nas abordagens dos conceitos introdutórios, existiu também a delimitação dos problemas gerados durante as disputas no jogo. Essa delimitação ocorreu com a seleção de problemas que envolvessem a movimentação do robô.

Os Gestores apresentaram os problemas às equipes dos Torneios, sem instrução prévia de sua resolução, seguindo a base da metodologia PBL. Os Gestores tiveram o papel de auxiliar na delimitação dos problemas identificados e instigar equipes a buscar os conceitos relacionados com hipóteses de soluções. A Tabela 17 exibe os eventos do Robocode selecionados para caracterizar os problemas apresentados as equipes. As hipóteses de soluções dos problemas listados na Tabela 17 puderam admitir os conceitos introdutórios de linguagem de programação, delimitados no presente trabalho.

Tabela 17 - Eventos Definidos como Problemas de Movimentação.

Evento	Característica do Problema (PBL)	Consequências do Problema	Hipóteses
run()	Problemas de movimentação podem ser identificados logo no início de cada *round*. Estes problemas podem ser relacionados ao principal evento do Robocode, *run()*.	Enquanto nenhum outro evento for executado, a programação do evento *run()* executa uma estratégia de movimentação do robô desfavorável (ex. movimentações curtas com método em sentido único).	Análises dos adversários, combinações de métodos que permitam locomoção em vários sentidos: girar robô para esquerda ou direita em graus; movimentos frente ou traz com distância em pixel definida.
onHitWall()	Um problema de movimentação pode ser percebido quando o robô colide excessivas vezes na parede da arena.	Excessivas colisões na parede. Impacta na perda de energia do robô a cada incidência. O robô pode perder uma partida apenas com colisões.	Quando identificar colisão na parede, o código deve estar preparado para mudar a direção do robô em graus para evitar a incidência (*turnLeft(180)*).
onHitRobot ()	Problema de colisão com outro robô pode surgir quando o radar não identifica a existência de adversários. Pode ocorrer quando existe uma falha na coleta dos dados do ambiente.	Colisões com outros robôs resultam na perda de energia equivalente a um disparo e devem ser evitadas. Nas 5 primeiras rodadas, quando todos os robôs estão na arena, sua probabilidade aumenta.	Pode utilizar mesma tática da colisão na parede (*turnLeft(180)*). Equipe pode criar estratégias diferentes e aproveitar a chance de proximidade para efetuar disparos.

Em virtude das regras descritas no edital, as equipes deveriam entrar nos torneios com propostas de definições de duas estratégias diferentes ao longo das disputas. Esse fato ocorreu por conta da divisão dos Torneios em primeira e segunda etapa. Na primeira etapa do Torneio, o jogo foi disputado com os robôs na modalidade "Todos Contra Todos" (todos robôs simultâneos em disputas), que exigiu uma estratégia de movimentação que permitisse sobreviver e estar sempre entre os primeiros colocados na arena (consiste em uma etapa classificatória). Na segunda etapa do Torneio, a estratégia deveria mudar, pois são confrontos diretos, "Um Contra Um". Os confrontos diretos, em geral, deveriam adotar uma estratégia que pudesse decidir o quanto antes a disputa. Analisando especificamente o problema da movimentação, quanto mais tempo de disputa, maiores foram as probabilidades de colisões e, consequentemente, perda de energia.

Em instituições de ensino que oferecem cursos técnicos ou superiores na área de informática, é comum para alunos ingressantes o primeiro contato com disciplinas que envolvam programação de computadores logo nos primeiros meses. Professores que ministram estas disciplinas devem, antecipadamente, preparar planos de ensino e aulas a fim de distribuir e balancear corretamente os conteúdos, conforme a carga horária da disciplina. Conteúdos introdutórios de linguagem ou conceitos de programação, em geral, são apresentados logo no início das disciplinas relacionadas à programação. A definição dos problemas apresentados pelos gestores às equipes teve por objetivo atender a conceitos introdutórios da linguagem de programação presentes nos planos de ensino/aula. Na Figura 44 (a), observa-se a aplicação de conceitos presentes nos planos de ensino da disciplina de linguagem de programação, plano de aula de conteúdo programático referente aos "comandos de decisões" *if-else* (se-senão). A Figura 44 (a) exibe o código de uma equipe do curso técnico integrado em informática com noções ou conhecimento superficial da linguagem de programação Java. Essa equipe propôs "variáveis booleanas" (ângulo==true) com instruções para "tomada de decisão" (*if-else*) quanto à movimentação do robô. O exemplo indica *if* (se) o ângulo é verdadeiro, o robô deve avançar o valor de 100 e virar à direita em 90 graus, *else* (senão) ângulo é falso e o robô deve recuar o valor de 50 e virar para esquerda em 90 graus (Meira et al., 2016).

```
5  public void run() {                    (A)
6      boolean angulo = true;
7  while(true) {
8      if (angulo == true){
9          ahead(100); turnRight(90);
10         angulo = false;}
11     else{
12         back(50); turnLeft(90);
13         angulo = true;}
14     } }
```

```
16 public void run() {                    (B)
17 while(true) {
18     for (int virarDir=0;virarDir<4;virarDir++){
19         ahead(100);
20         turnRight(90);}
21     for (int virarEsq=4;virarEsq>0;virarEsq--){
22         ahead(100);
23         turnLeft(90);}
24         }
25     }
```

Figura 44 - (a) Comandos de Decisão (b) Estruturas de Repetição (Meira et al., 2016).

Os códigos exibidos na Figura 44 (a) e (b) foram desenvolvidos pela mesma equipe em momentos distintos dentro da competição. O Código (b) foi criado em rodadas posteriores ao (a). A Figura 44 (b) indica o uso do conceito de "estrutura de repetição" com comando *for* (para), apresentado na linha 18. Na mesma linha, observa-se o uso do comando for com sua "variável de inicialização" (int), as "condicionais" recebendo valores (inicial: virarDir=0; final: virarDir<4) e ao final seu "incremento" (virarDir++). Para entender o conceito, a equipe pesquisou informações dos conceitos envolvidos acerca da "variável de inicialização" (inicia um controle de repetição), as "condições" (determina a condição inicial e final da repetição) e "incremento" (variável de controle alterada a cada repetição). Após o entendimento dos conceitos, a equipe pôde aferir o resultado em comandos de movimentação do robô: avançar no valor de 100 e virar à direita em ângulo de 90 graus por 4 vezes consecutivas; em seguida repetir o mesmo processo à esquerda (Meira et al., 2016).

Muitas equipes conquistaram melhorias significativas em relação as estratégias de movimentação programadas nos robôs. Puderam atingir estágios que envolveram temas como a predição de movimentos lineares e previsão de movimentos para disparos. A predição de movimentos lineares, exibida na Figura 45 (a), pôde ser programada a partir da posição do adversário com auxílio de fórmulas matemáticas. A previsão de disparos, assim como a predição de movimentos lineares, pôde ser obtida a partir da posição do adversário. Exibida na Figura 45 (b), a previsão tem sua base na identificação de movimentação, direção e velocidade do adversário. Ocorreu na execução da programação de cálculos matemáticos com objetivo de predizer o momento exato do disparo, levando-se em consideração: ângulo; tempo estimado para bala atingir o alvo e distância. A Figura 45 (b) indica o robô que se deslocou do ponto A até o ponto B sobre uma circunferência de centro O e raio R. O comprimento ΔS do arco (AB) caracteriza o espaço percorrido pelo robô, e o ângulo central $\Delta\Theta$ oposto ao arco (AB) é o deslocamento angular ($\Delta\theta = \theta B - \theta A$). De acordo com a Figura 45 (b), em um intervalo de

tempo Δt, o robô em movimento circular executou um deslocamento angular Δθ. A velocidade angular média (ωm) do robô nesse intervalo de tempo pode ser definida pela fórmula apresentada na Figura 45 (b).

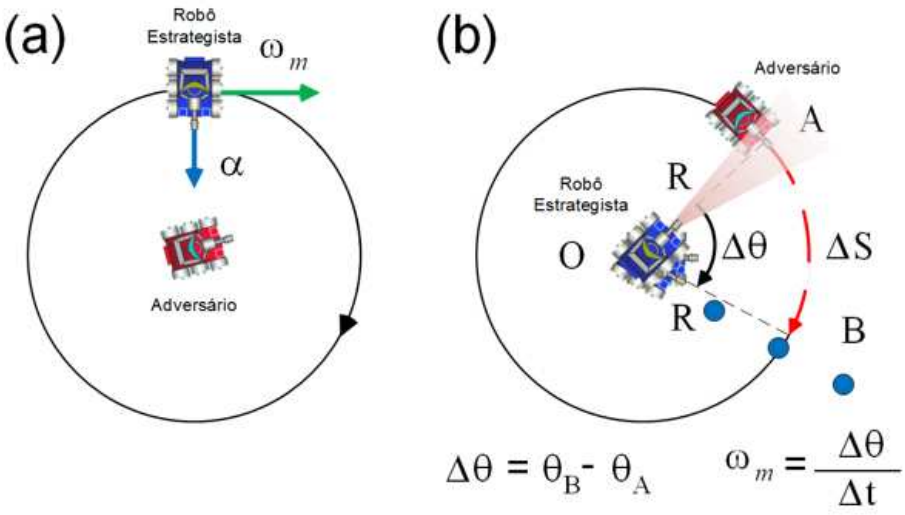


Figura 45 - Movimentos de Predição e Previsão.

6.3. Pesquisa de Campo Robocode Brasil

Por questões de localização geográfica entre instituições e tempo hábil para coleta de dados durante a ocorrência das primeiras rodadas dos Torneios Locais, optou-se por selecionar duas instituições. Foram selecionadas as instituições Anhanguera e IFSP por apresentarem características distintas. A Anhanguera enquadrou-se como uma instituição estreante nos torneios do LIAG, pois nunca havia se envolvido com quaisquer outras ações relacionadas ao jogo Robocode. Por outro lado, o IFSP foi caracterizado como uma instituição veterana em ações com Robocode, com participação na Liga promovida no LIAG-UNICAMP (2015), Olimpíadas Geek[29] (2016 e 2015) organização de Torneios Locais (2014 e 2011) nas semanas de Ciência e Tecnologia e Científico Cultural respectivamente. Outra característica distinta entre as instituições deveu-se ao fato da Anhanguera contar com alunos do curso superior de Bacharelado em Ciência da Computação enquanto o IFSP contou com alunos do curso Técnico

[29] Olimpíadas Geek: competições realizadas anualmente no IFSP – Capivari com disputas de várias modalidades de jogos digitais e educacionais, dentre disputas com Scratch e Robocode.

Integrado em Informática. A Tabela 18 mostra informações sobre as instituições Anhanguera e IFSP, participantes do Robocode Brasil 2016.

A Faculdade Anhanguera de Rio Claro-SP e o IFSP – Capivari-SP foram selecionadas durante a execução do Robocode Brasil como Torneios representantes para coleta de dados da documentação direta. As bases da documentação direta estão relacionadas no levantamento de dados no próprio local onde os fenômenos ocorrem (Marconi & Lakatos, 2010). Estes foram obtidos com pesquisa de campo na conquista de informações sobre a problemática envolvida, no caso do presente trabalho: se combinações de PBL, jogo educacional e competições científicas são motivadoras para aprendizagem de linguagem de programação.

Tabela 18 - Perfil de duas Instituições Participantes.

Participações / Instituição	**Participação em Eventos Anteriores do LIAG Robocode**	**Participação/Organização em Outros Eventos com Ambiente Robocode**	**Perfil dos participantes no Torneio Local Robocode Brasil 2016**
Faculdade Anhanguera Rio Claro	Nunca	Nunca	Superior do Curso Bacharelado em Ciência da Computação (participantes do 1º ao 4º ano)
IFSP Campus Capivari	Liga LIAG Robocode 2015	2014 – Torneio local semana de Ciência e Tecnologia; 2011 – Torneio local na Semana Científico Cultura; 2015 e 2016 – Olímpiadas Geek.	Técnico do curso Integrado em Informática (participantes do 1º e 3º ano)

Desconsiderando as características distintas de participações, experiências em competições com Robocode e níveis de cursos técnicos e superiores, as duas instituições apresentaram números muito próximos para primeira fase da pesquisa de campo. Esta consistiu no levantamento das variáveis da pesquisa de campo: (1) Verificação de hipóteses; (2) Adesão ao Torneio Local; (3) Motivações das instituições no Torneio Local.

Para verificação da hipótese da pesquisa, a primeira variável (verificação de hipóteses) da pesquisa de campo, ficou condicionada aos Gestores: a oferta de condições necessárias à promoção da PBL; jogos educacionais; e ambiente propício a realização da competição. A Tabela 19 demonstra itens que puderam representar motivações dos alunos em participar do Robocode Brasil. A impossibilidade de transmissão *on-line* dos torneios no IFSP, incidiu diretamente nos itens de horários e localização das disputas. Itens: transmissão *on-line* (por questões de infraestrutura); horários estipulados para disputas e localização impossibilitaram ou desestimularam alunos dos cursos superiores do período noturno que tinham interesse no torneio.

Tabela 19 - Itens para Motivar Alunos na Participação da Competição.

Variáveis / Instituição	Faculdade Anhanguera Rio Claro	IFSP Campus Capivari
Participação/Envolvimento Gestor Local	X	X
Envolvimento direto Coordenadores / Gerentes da Instituição	X	X
Disponibilização de espaço físico e infraestrutura para torneis	X	X
Localização/Horários/Facilidade de acesso para participações de todos curso de TI nos torneios	X	–
Divulgação do evento pelos professores da Instituição	X	X
Divulgação do evento em outros canais da Instituição	–	X
Transmissão *on-line* das disputas dos torneios	–	–
Comunicação dos Gestores Locais com Líderes e equipes	X	X
Estímulo juntos aos alunos para conhecer / participar dos torneios	X	X

Outra questão pertinente a pesquisa de campo foi indicada na segunda variável da pesquisa de campo, a adesão ao Torneio Local. Essa adesão foi medida através das inscrições dentre os cursos propostos para divulgação. A Figura 46 (a) mostra que Anhanguera possuía um total de 94 alunos regulamente matriculados divulgados no curso superior de Ciência da Computação, sendo que 26 motivaram-se em formar equipes e aderiram ao Torneio Local Robocode. A Figura 46 (b) mostra o IFSP em um total de 67 alunos regulamente matriculados divulgados no curso técnico Integrado em Informática, 22 inscreveram-se. A Figura 46 (c) posicionou, com cerca de 28%, a participação das 2 (duas) instituições somadas representaram no quantitativo global na competição.

A Figura 47 indica a terceira e última variável da pesquisa de campo, relacionado as motivações dos alunos participantes no Torneio Local. Foram consideradas Anhanguera e IFSP, respectivamente, e as principais motivações perguntadas e contabilizadas na pesquisa de campo realizada durante as primeiras rodadas dos Torneios Locais. Nas motivações recorrentes foi possível indicar: (1) Competição; (2) Aprendizado; (3) Professores e (4) Jogo. Dentre as 4 (quatro) principais incidências, 3 (três) motivações (competições, aprendizado e jogo) puderam ser diretamente relacionadas com a hipótese do trabalho em associar jogo e competições para afetar positivamente o resultado da aprendizagem.

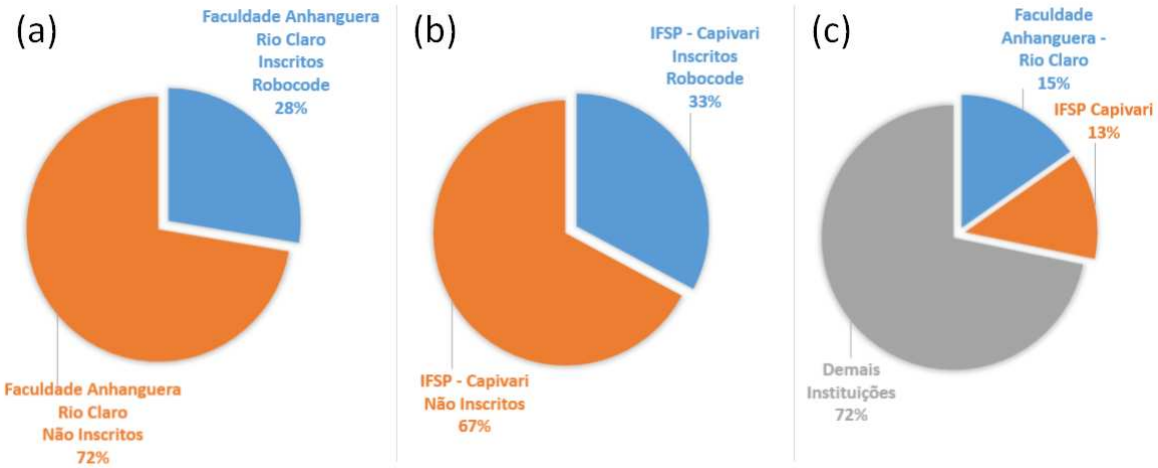


Figura 46 – Inscrições: (a) Anhanguera; (b) IFSP; (c) “a” e “b” na competição.

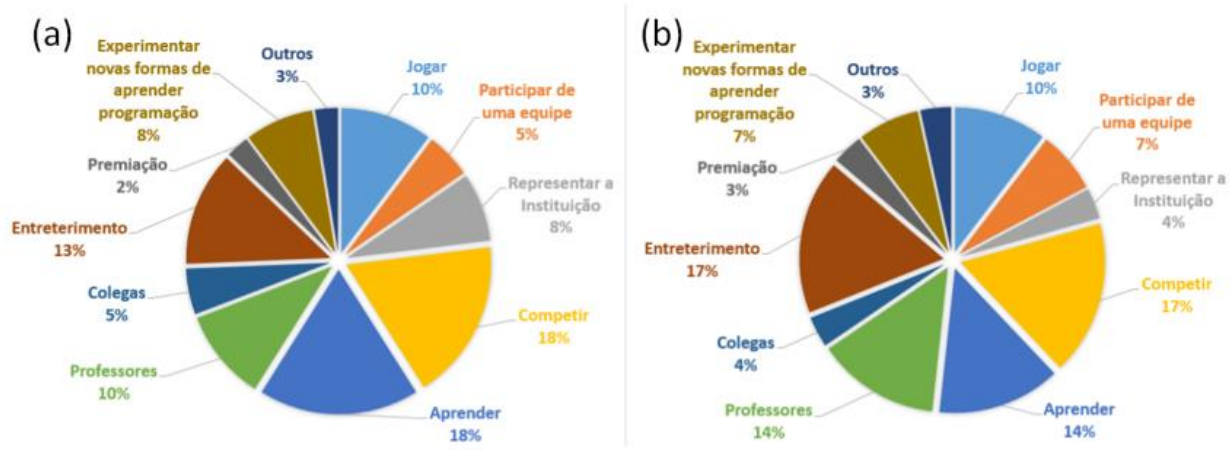


Figura 47 - Resultado das Motivações da Pesquisa de Campo.

6.4. Análise da experiência dos Gestores

Os gestores tiveram papel fundamental na organização e ocorrência dos Torneios nas instituições. Foram desenvolvidas entrevistas, pesquisa-ação e documentos com sugestões de ações relacionadas à PBL com objetivo de organizar e promover participação ativa dos Gestores. Algumas instituições que competiram na Liga LIAG de 2015, através de professores, mostraram interesse e colaboraram na construção de ações que deram origem ao Robocode Brasil 2016. As próximas subseções apresentam características da equipe mobilizada nas ações do Robocode.

6.4.1. Entrevistas com Gestores

As entrevistas foram realizadas no final do ano de 2015 e início de 2016, com objetivo de elencar informações que pudessem caracterizar situações ou problemas vivenciados na competição de 2015. O tipo de entrevista selecionado foi despadronizada ou não estruturada, na qual tanto o entrevistado quanto entrevistador têm liberdade de desenvolver as situações de modo que considere adequado. A entrevista seguiu bases da observação direta intensiva com objetivos na descoberta de planos de ação[30] e na conduta atual[31].

"A entrevista tem como objetivo principal a obtenção de informações do entrevistado, sobre um determinado assunto ou problema" (Marconi & Lakatos, 2010). Com este propósito, o foco da entrevista restringiu-se à Liga LIAG Robocode realizada no ano de 2015. Convites foram realizados aos professores que organizaram e orientaram alunos das seguintes instituições participantes da Liga de 2015: (1) IF Farroupilha; (2) IFSP – Hortolândia; (3) IFSP – Capivari.

A instituição IF Farroupilha, foi campeã da Liga LIAG Robocode 2015. No IF Farroupilha, além do professor orientador e gestor do Torneio Local, participaram da entrevista alunos integrantes da equipe campeã da competição. A entrevista foi realizada por videoconferência, ilustrada na Figura 48, entre Administração da Competição e o IF Farroupilha do campus da cidade de São Borja-RS. A equipe do IFSP do Campus de Hortolândia-SP conquistou o Torneio Local do IFSP e ficou em 4ª (quarta) colocação geral na competição. Participou da entrevista uma professora que ministra disciplinas de robótica e linguagem de programação, responsável por orientar as equipes da competição Robocode do IFSP de Hortolândia. Um professor da área de programação que ministra disciplinas no IFSP Campus de Capivari-SP e na Faculdade Anhanguera de Santa Bárbara d'Oeste-SP representou o entrevistado do IFSP – Capivari. Este professor colaborou com o Torneio do IFSP e mostrou interesse na organização de competições durante o ano de 2016 na Anhanguera de Santa Bárbara d'Oeste-SP.

[30] Descoberta de planos de ação. Descoberta por meio de definições individuais qual a conduta adequada em determinadas situações.
[31] Conduta atual. Inferir que conduta a pessoa terá no futuro, conhecendo a maneira pela qual ela se comportou no passado ou se comporta no presente, em determinadas situações.

Figura 48 - Entrevista em Videoconferência com IF Farroupilha.

O resultado caracterizou o registro dos objetivos da entrevista para determinação de uma base de dados com pontos fortes e fracos que pudessem auxiliar no desenvolvimento do Robocode Brasil 2016. A associação dos objetivos da entrevista resultou na Figura 49 sob a forma de análise SWOT[32] da Liga LIAG de 2015. A partir da análise SWOT foi possível levantar os principais pontos a serem tratados na pesquisa-ação.

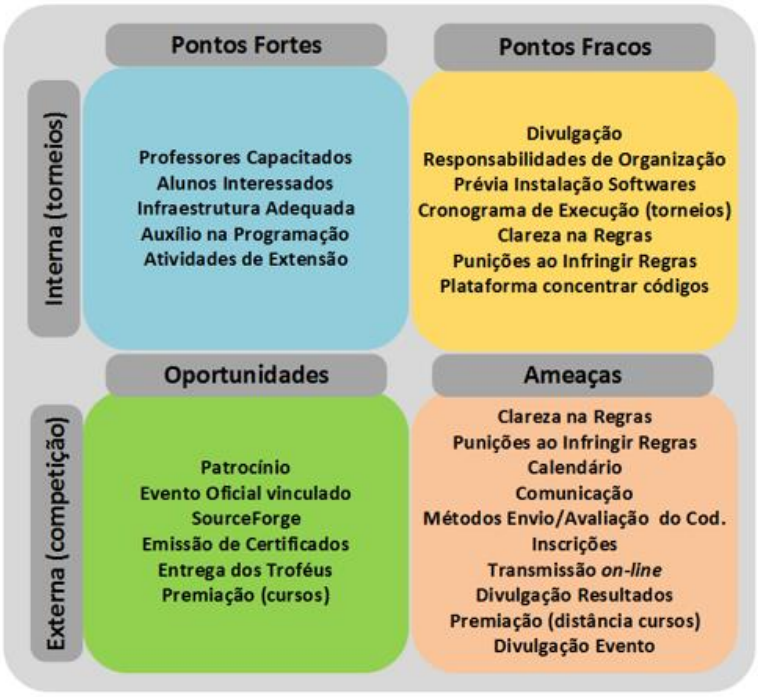


Figura 49 - Análise SWOT Liga LIAG Robocode 2015.

[32] SWOT, é uma sigla que significa *Strenghts* (Forças), *Weaknesses* (Fraquezas), *Opportunities* (Oportunidades) e *Threats* (Ameaças).

6.4.2. Pesquisa-ação para Robocode Brasil

A pesquisa-ação foi eleita como forma de compreensão das situações encontradas nas entrevistas, da seleção dos problemas bem como a busca de soluções. De posse dos dados extraídos das entrevistas, a pesquisa-ação manteve a mesma equipe das entrevistas. Essa equipe trabalhou em reflexões e críticas de modo cooperativo sobre o conjunto de problemas detectados na competição de 2015.

Tabela 20 - Identificação e Resolução de Problemas nas Competições 2015/2016.

Problema Identificado	Como era tratado 2015	Solução Proposta 2016
Responsabilidades de Organização torneios locais	Não existia limites estabelecidos de responsabilidades	Foi criado o Gestor do Torneio local. Em geral um professor da Instituição
Clareza nas Regras	Abstrata. Margens para interpretações dúbias ou inexistentes.	Desenvolvimento de um edital completo de execução com item regras.
Cronograma de execução	Macro. Informado no site	Micro, detalhado em itens especificados em edital.
Calendário das disputas	Aberto. Delimitações iniciais e finais informados no site.	Fechado. Detalhado semanalmente especificados em edital.
Punições ao infringir regras	–	Conjunto de regras especificados em edital.
Inscrições	E-mail	Automatizado no Gerenciador da competição. Inscrições especificadas em edital.
Plataforma para receber códigos	–	Automatizado no Gerenciador da competição. Regras de envio especificadas em edital.
Divulgação	Canais descentralizados de divulgação	Disponibilização das informações em domínios oficiais http://www.robocodebrasil.com.br e canais de redes sociais. Planos para divulgação de torneios e liga com antecedência de 6 meses.
Divulgação dos Resultados	Manual, lento.	Automatizado no Gerenciador. Ao final de cada disputa pode ser gerado pelo gestor *on-line*.
Instalações de *Softwares*	–	*Check list* enviado ao gestor com *softwares* necessários para criar torneios e garantir acesso para uso dos alunos.
Comunicação	E-mails particulares	Estruturação comunicação: e-mails oficiais @robocodebrasil.com.br *Newsletter*, redes sociais, *chats*, videoconferências etc.
Avaliação dos códigos	Apenas nas etapas finais	Em todos momentos da competição. Torneios a cargo dos gestores. Liga a cargo dos administradores Robocode.
Transmissão on-line	–	Todas disputas do Torneio Público e Liga transmitidos *on-line*
Premiação	Brindes e cursos	Aumento no valor dos brindes e abolição dos cursos para liga. Inviabilização de participações de instituições geograficamente distantes

A pesquisa-ação também teve por objetivo a criação de processos inexistentes, definidos a partir da demanda de um conjunto de novas atividades apresentadas e pertinentes a competição. Os resultados das entrevistas foram importados para pesquisa-ação com o nome de "Aperfeiçoamento da Competição Robocode". Professores envolvidos na pesquisa-ação identificaram os problemas e apresentaram propostas de soluções que foram analisadas e discutidas em grupo. A Tabela 20 apresenta os problemas identificados na competição de 2015 e sua conclusão, com as soluções propostas trabalhadas em grupo. Todas propostas de soluções apresentadas foram executadas no Robocode Brasil 2016.

6.4.3. Orientações aos Gestores do Robocode Brasil

Com o fechamento das entrevistas e execução da pesquisa-ação na competição Robocode Brasil 2016, pôde-se receber as inscrições por intermédio das equipes e professores das instituições. Professores, em geral, de instituições devidamente validadas, assumiram o papel de Gestores dos Torneios Locais dentro da competição Robocode Brasil 2016. As ações levantadas, na pesquisa-ação, apoiaram a continuidade da participação dos professores para auxiliar a estruturação do ambiente da competição. Neste contexto, o envolvimento dos professores (estendendo a colaboração para funcionários em geral da instituição) tiveram uma importante parcela no acompanhamento das equipes durante a competição.

Os professores tiveram um ambiente de competição com a possibilidade de motivar alunos e colaborar na estruturação de conceitos teóricos/práticos de linguagem de programação e suas relações com programações do robô e, consequentemente, nas estratégias das disputas. Professores que fizeram uso destas características, refletiram na potencialização dos aspectos organizacionais inerentes à competição como: (1) Definições de metas de jogo; (2) Pesquisa de conceitos; (3) Propostas de nivelamento entre equipes; (4) Incentivo a busca contínua no aperfeiçoamento dos robôs; (5) Discussões de estratégias a cada nova rodada; (6) Acompanhar a participação de cada equipe.

As orientações foram desenvolvidas pela Administração da Competição e tiveram por objetivo auxiliar os Gestores dos Torneios a detectarem os desafios gerados durante as disputas no jogo, que são os problemas, e auxiliar as equipes a traçar estratégias de soluções com base no PBL. Um dos indícios, propositalmente planejados para atender o PBL, foi a divisão dos

alunos em pequenos grupos que deveriam desconstruir e analisar problemas. Professores deveriam auxiliar as equipes a ordenar os problemas de acordo com conceitos relacionados à disciplina de linguagem de programação. Nos Torneios, os problemas são pautados conforme condições apresentadas durante o jogo, na execução de cada rodada. Os detalhes das orientações são descritos a seguir.

De acordo com a metodologia PBL, um problema denominado "Base" foi criado pela Administração da Competição como parte integrante das orientações. Esse problema Base deveria oferecer subsídios para que professores e alunos orientem-se, para adaptá-los a sua realidade de desafios na competição. Foi esperado do problema Base um suporte de referência para gerar interações com o propósito de analisar e resolver situações de jogo (por exemplo, problemas de movimentação do robô). O problema Base serviu também para suportar a introdução de novos conceitos de linguagem de programação para desenvolvimento dos problemas reais ou simulados da competição.

A elaboração do problema Base foi vinculada a introdução a programação de robôs com Robocode. O problema foi descrito na sequência das ocorrências: (1) O código *default* (conhecido como código base em Java) é apresentado sempre que um novo robô é criado (Figura 50); (2) No código *default*, observação na sequência da estrutura de movimentação o evento *onHitWall* (quando bater na parede); (3) O evento *onHitWall*, executado quando o robô bate na parede; (4) Voltar, *back(20)* configurado com parâmetro de valor 20 (vinte) pixels; (5) O primeiro desafio foi apresentado: desenvolver estratégias para quando o robô bater na parede.

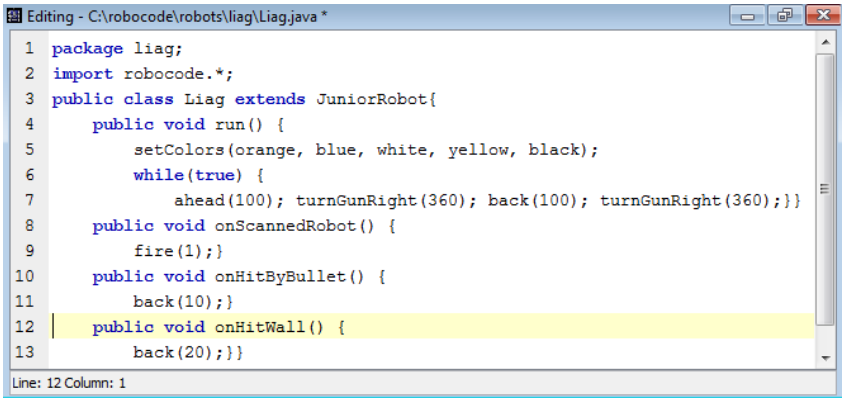


Figura 50 - Código Default do Robô.

As características de Aprendizagem por Design (LBD) foram importantes para desenvolver alguns elementos de práticas pedagógicas com objetivo de gerar um

sequenciamento estrutural com a seleção de elementos necessários para construção do problema (Tabela 21) (Assis, 2011; Evans & Nation, 2000; Koper & Tattersall, 2005).

Tabela 21 - Estrutura do Problema com Base LBD.

Elementos LBD	Descrição
Título	Robô bate na parede
Objetivos de Aprendizagem	Introdução a programação de métodos.
Pré-requisitos	Conclusão da disciplina de Lógica de Programação
Componentes	Atividades de pesquisa para programação de robôs no Robocode. Equipe de 4 alunos.
Métodos	PBL no Robocode, *Learning by Design* (LBD).

A apresentação de um problema incompleto segue uma das características marcantes do PBL em apresentar conceitos necessários para sua resolução (Filho & Ribeiro, 2009). Seguem os passos gerais do PBL com: a discussão do problema; investigação do que sabem; geração de hipóteses; obtenção dos objetivos de aprendizagem e atribuições de tarefas formatadas na Tabela 22 (Hou, 2014).

Com a abordagem *Learning By Design* (LBD), os alunos foram submetidos aos desafios do problema incompleto que puderam envolver: (1) Pesquisas de casos relacionados, como estudar o comportamento de outros robôs quando batem na parede; (2) Raciocínio lógico, ao bater na parede quais medidas poderão ser tomadas; (3) Conhecimento prévio, com base na análise de disputas anteriores e o próprio conhecimento de conceitos de programação, para apontar proposta de solução ao problema. O problema apresentado na Figura 51, chamado de problema incompleto no PBL, exigiu dos alunos uma pesquisa para aprofundar os conhecimentos incompletos das regras e características do Robocode.

Durante os desafios, professores puderam disponibilizar fontes de recursos para as pesquisas de acordo com o planejamento pedagógico (Kolodner et al., 1996). Perguntas relacionadas ao conhecimento prévio do aluno puderam ser instigadas: “O que você faz em um jogo convencional quando não sabe a informação de como passar de fase?” O desafio no âmbito do levantamento e discussão de informações para resolução do problema é apresentado da Tabela 22.

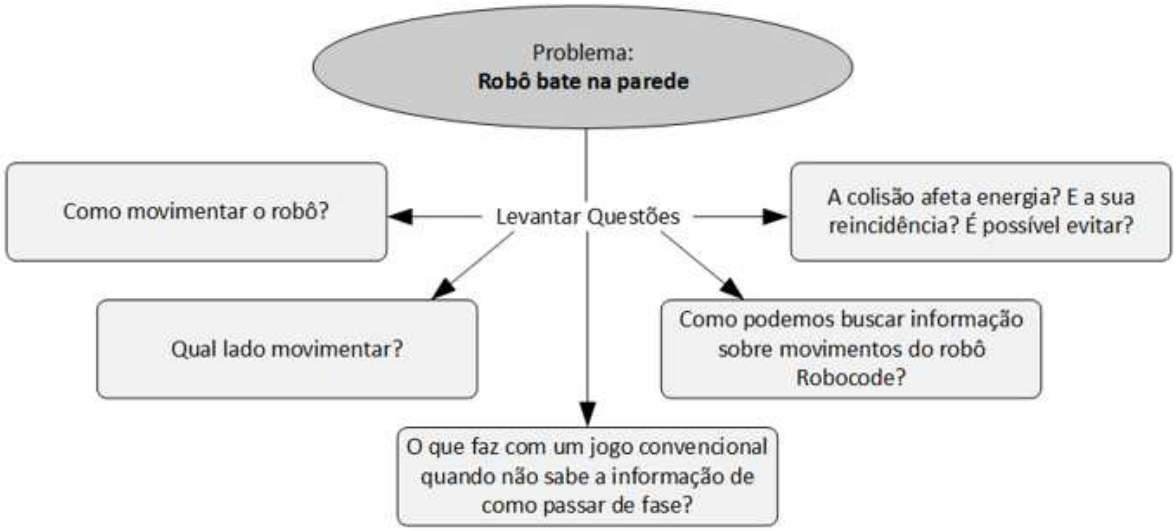


Figura 51 – Problema de Movimentação da Proposta.

Tabela 22 - Dinâmicas para Levantamento de informações.

Desafio	Característica na Resolução do Problema
1 – Pesquisar em Revistas especializadas	Aluno chega a conclusão, assim como em jogos convencionais, manuais técnicos e artigos acadêmicos são ótimas fontes para encontrar informações do ambiente de pesquisa.
2 – Classificar Informação	Após busca de artigos, manuais etc. realização do processo de triagem e classificação de informações por grau de pertinência em resposta ao problema apresentado.
3 – Comentar o artigo	Ajudou a encontrar a movimentação do robô? É a único modo de resolver o problema? Achou alguma informação relevante além do problema? Comentar o artigo com base na pergunta com intuito de montar uma base de dados de informações a se considerar junto ao Robocode.
4 – Reunião em grupo	Reunir em grupo para apresentar a pesquisa e informações pertinentes levantadas por cada integrante a fim de elencar o que será aplicado na resolução do problema. Faz se necessário um *brainstorm* para registro de ideias.
5 – Sequencia para resolução do problema	Descrever os passos necessários para solucionar o problema. Discutir com demais grupos as soluções apresentadas. Exibir a resolução e justificar o motivo da sequência adotada.

A Tabela 22 utilizou características do segundo princípio da abordagem LBD proposto por Gee (2013), Aprendizagem Baseada em Bons Problemas (*Good Problem Based Learn*): utilização de fatos e informações necessárias para resolução (*Problem Solvem*); ordenação de problemas de forma a conduzir os jogadores a formularem hipóteses que funcionem na resolução (*Well-ordered Problems*); estabelecimento de uma sequência de resolução (*Sequencing*) que pode ser observada na Figura 52.

Para resolução do problema, além dos benefícios diretos em pesquisar programação de métodos (métodos *ahead*, *back*, *turnRight*, *turnLeft*, ex. *getBearing()* - ângulo em graus da parede batida em relação ao seu robô.), indiretamente tiveram acesso as noções de programação de eventos (evento *onHitWall*) que dispara o método "quando robô bater na parede". A Figura 52 compilou algumas respostas de equipes para o problema apresentado.

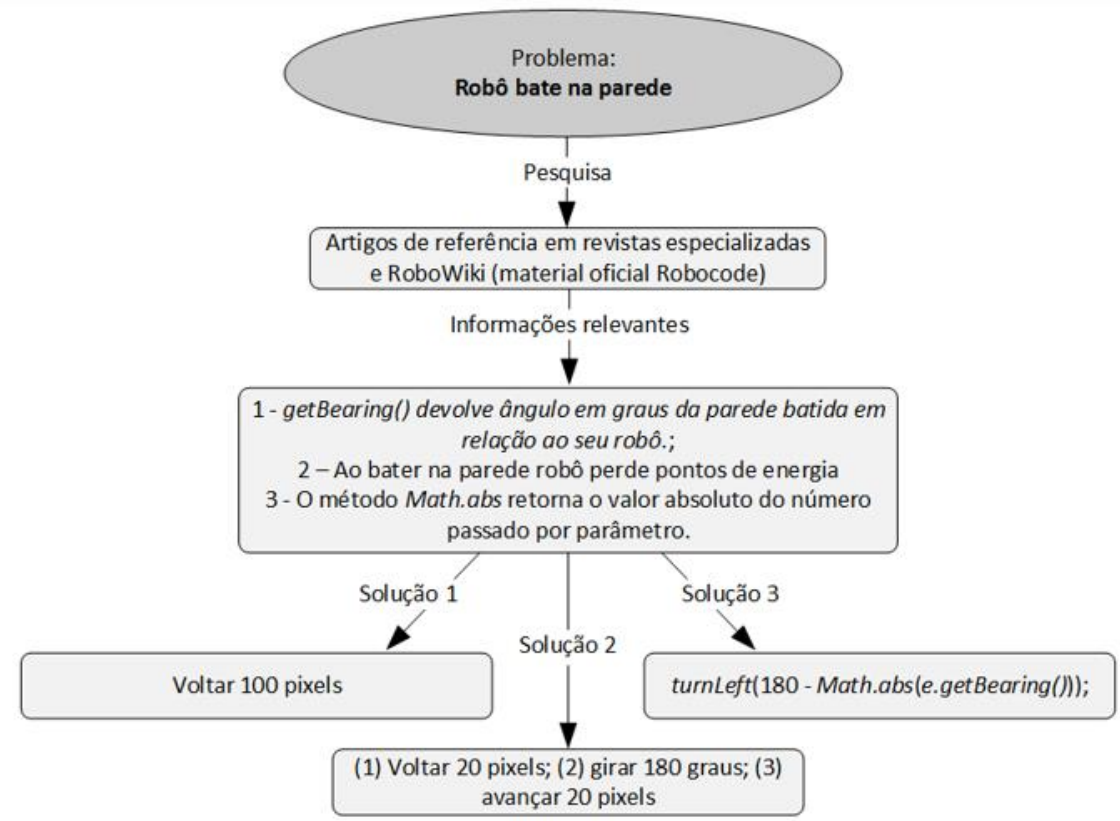


Figura 52 - Respostas ao Problema.

As resoluções dos problemas Base permitiram às equipes desenvolverem estratégias de movimentação quando o robô bater na parede e exibiu a necessidade de registrar as regras para desenvolvimento do código, conforme as condições apresentadas durante a disputa. A Figura 53 sugere estratégias para desenvolver a movimentação do robô quando bater na parede. Os assuntos envolvidos focaram estruturas introdutórias da ementa da disciplina de linguagem de programação. O problema Base seguiu os conceitos apresentados por Berbel (1998) para PBL, com envolvimento em fases para resolução. Na primeira fase, os alunos formularam os objetivos de aprendizagem a partir da discussão do problema. Na segunda fase, após a realização do estudo, os alunos rediscutiram o problema com base nos novos conhecimentos adquiridos.

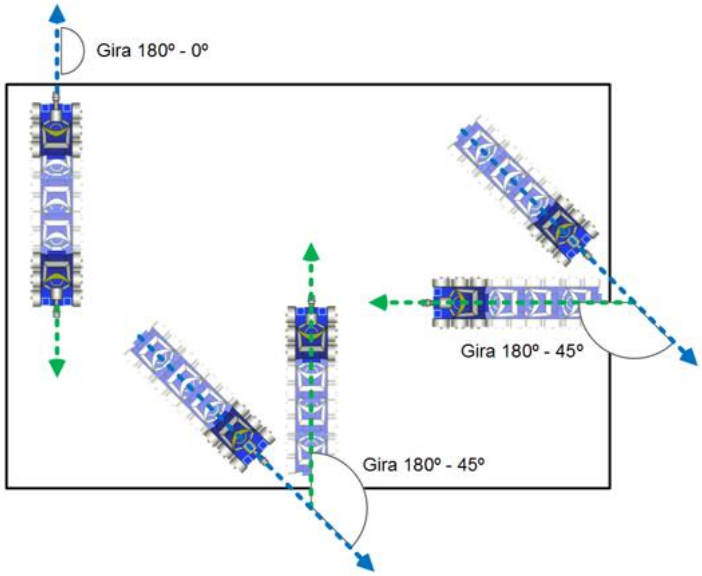


Figura 53 - Estratégias de Movimentação do Robô.

O problema Base apresentado (Figura 51) admite várias respostas corretas (Figura 52). Enquadram-se em características de desenvolvimento de bons problemas PBL (Duch, 2001), com problemas abertos para comportar várias respostas igualmente válidas. Coube aos alunos analisar situações de resolução dos problemas no Robocode e criar estratégias para definirem melhores soluções. A PBL trata de uma estratégia pedagógica em que o aluno deve formular seus objetivos de aprendizagem e estudar para aprofundar conhecimentos incompletos a partir das hipóteses levantadas.

6.5. Análise da experiência dos Alunos

Um dos objetivos deste trabalho foi avaliar se competições científicas com Robocode estimularam aprendizagem de conceitos de linguagem de programação em cursos técnicos e superiores da área de informática. Para isso, foram desenvolvidos 2 (dois) questionários para analisar a experiência dos alunos em dois momentos distintos da competição Robocode Brasil 2016. Os questionários foram denominados de inicial e final (Apêndice VI e Apêndice VII respectivamente) tratados nas próximas subseções de seção. A ferramenta *on-line* de criação e gerenciamento de formulários da Google® foi selecionada para os questionários; solução *on-line* TIBCO® SpotFire foi utilizada no auxílio ao desenvolvimento da análise e visualização dos dados. Os canais oficiais do Robocode Brasil foram utilizados para compartilhar os

questionários. Os questionários estiveram disponíveis para todas instituições participantes do Robocode Brasil 2016. Notas textuais foram criadas para convidar os participantes a colaborar com o trabalho.

Um fator considerado como resultado positivo junto as equipes participantes foi a baixa evasão dos Torneios Locais e Público antes e durante a competição científica Robocode Brasil 2016. De acordo com informações consultadas no banco de dados do gerenciador da competição: apenas 3,7% das equipes não participaram de nenhuma rodada; apenas 5,5% das equipes desistiram no início das primeiras rodadas, da qual configurou um índice de evasão total baixo de 9,2%.

O presente trabalho foi submetido e aprovado pelo Comitê de Ética em Pesquisa (CEP) da UNICAMP – Campus Campinas-SP. O Apêndice VIII apresenta o parecer consubstanciado do CEP. No parecer constam o Certificado de Apresentação para Apreciação Ética (CAAE), número do parecer e os critérios que envolvem a pesquisa. Um dos principais critérios refere-se a apresentação de consentimento livre e esclarecido, quando se tratar de participantes caracterizados como população vulnerável (menores de idade). O modelo do Termo de Consentimento Livre e Esclarecido (TLCE) é apresentado no Apêndice IX. De posse do número do parecer e CAAE é possível consultar o projeto aprovado na íntegra na Plataforma Brasil[33].

6.5.1. Grupo de participantes

O questionário inicial foi desenvolvido e disponibilizado aos alunos participantes antes do início dos Torneios Locais/Público durante as primeiras rodadas. O questionário final foi aplicado nas últimas rodadas e final dos Torneios locais/público. Os Questionários Inicial e Final foram aplicados na competição Robocode Brasil 2016. As primeiras questões, presentes no questionário inicial, referiram-se às características gerais dos participantes da competição Robocode Brasil 2016. A Tabela 23 apresenta as "Questões Gerais" comuns ao grupo de participantes dos Torneios ("idG" representa a identificação das Questões Gerais, ex. "idG-1" refere-se a primeira questão geral).

[33] Plataforma Brasil consiste no sistema eletrônico do Governo Federal para sistematizar o recebimento dos projetos de pesquisa que envolvam seres humanos nos comitês de ética no Brasil.

Tabela 23 - Questões Gerais Robocode Brasil.

idG	Questão	Classificação / Objetivo	Objetivo
1	Qual sua Instituição?	Múltipla escolha – Mostruário (admite uma resposta)	Pergunta fato
2	Qual curso e período?	Pergunta Aberta	Pergunta fato
3	Qual gênero?	Múltipla escolha – Mostruário (admite uma resposta)	Pergunta fato
4	Qual sua idade?	Múltipla escolha – Mostruário (admite uma resposta)	Pergunta fato
5	Qual a renda familiar:	Múltipla escolha – Mostruário (admite uma resposta)	Pergunta fato

As Questões Gerais foram incorporadas com objetivo de traçar o perfil geral dos alunos participantes da competição Robocode Brasil 2016. Cerca de 100 (cem) alunos responderam as Questões Gerais. Para pergunta de "idG-1", observa-se na Figura 54 a porcentagem de participação dos alunos por instituição de ensino. A Instituição com maior porcentagem de participações nas respostas dos questionários foi o IF Farroupilha – Campus São Borja do estado de Rio Grande do Sul. Essa mesma instituição foi campeã da Liga LIAG 2015.

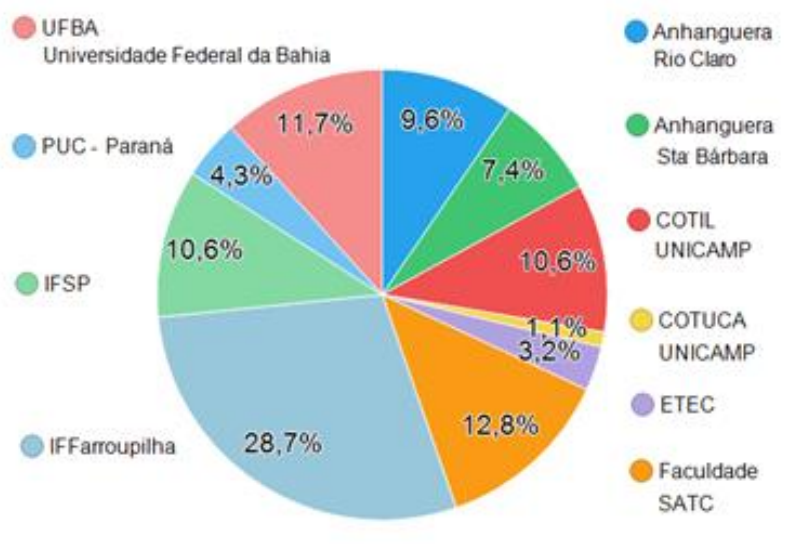


Figura 54 - Participação por Instituição Robocode Brasil 2016.

A pergunta de "idG-2" relaciona a incidência de cursos, nível e tipo de curso das quais participam os alunos inscritos na competição. A Figura 55 (a) mostra os cursos dos alunos inscritos na competição. Mais de 50% dos alunos participantes da competição foram dos cursos Técnicos Integrados em Informática e cursos superiores de Bacharelado em Ciência da

Computação somados, indicados na Figura 55 (a) nas cores laranja (27,7%) e azul (27,7%). É possível observar a incidência do curso de Tecnologia em Jogos Digitais dos alunos da PUC de Londrina-PR. A Figura 55 (b) apresenta o nível do curso, com 57,4% de nível superior e 42,6 de nível técnico/técnico integrado. A Figura 55 (c) exibe o tipo de curso com destaques para Bacharelado, 40,4% (Ciência da Computação e Sistemas de Informação), e Técnico Integrado, 24,5% (Técnico Integrado em Informática).

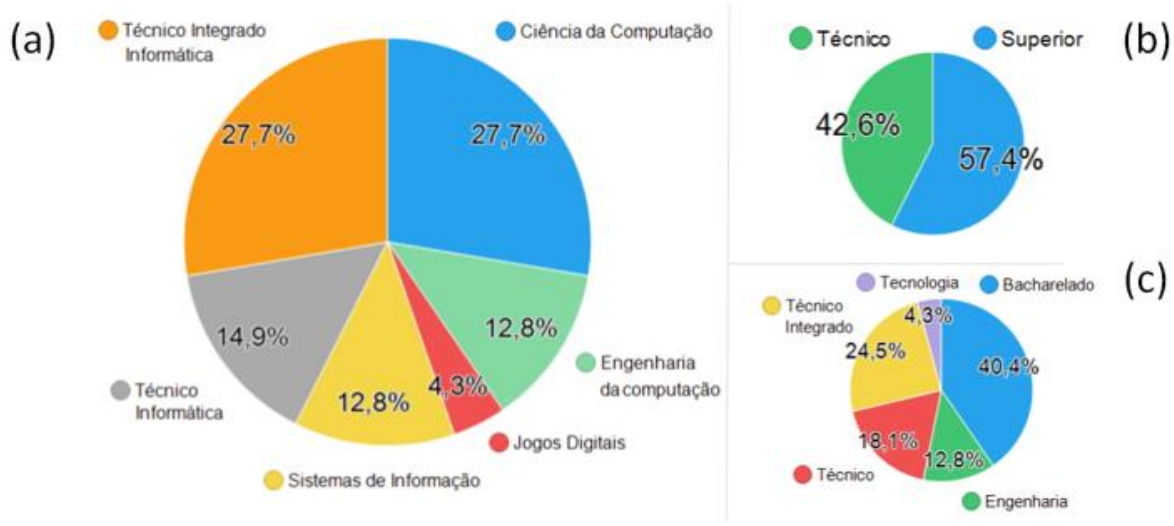


Figura 55 - (a) Curso; (b) Nível; (c) Tipo de Curso.

A Figura 56 (a) exibe a faixa etária dos participantes. É possível observar que mais de 41% dos alunos inscritos estão entre 15 e 18 anos, seguidos por 28,7% com idades entre 19 e 22 anos. Portanto, o público participante da competição foi representado por mais de 70% de jovens com idades entre 15 e 22 anos ("idG-3"). As demais faixas etárias não atingiram 30% do total de inscritos. A questão "idG-4", apresentada na Figura 56 (b), indica que 84% dos participantes são do gênero masculino enquanto apenas 14,9% representantes do gênero feminino. Com exceção dos cursos técnicos integrados em informática, que apresentam turmas equivalentes no quesito gênero, os demais cursos são predominantemente compostos por alunos do gênero masculino. Contudo, mesmo para cursos técnicos integrados em informática, a adesão na competição científica Robocode 2016 mostrou-se predominantemente de alunos do gênero masculino. Para questão "idG-5", de renda familiar, Figura 56 (c), apresentou-se predominantemente alunos com baixa renda.

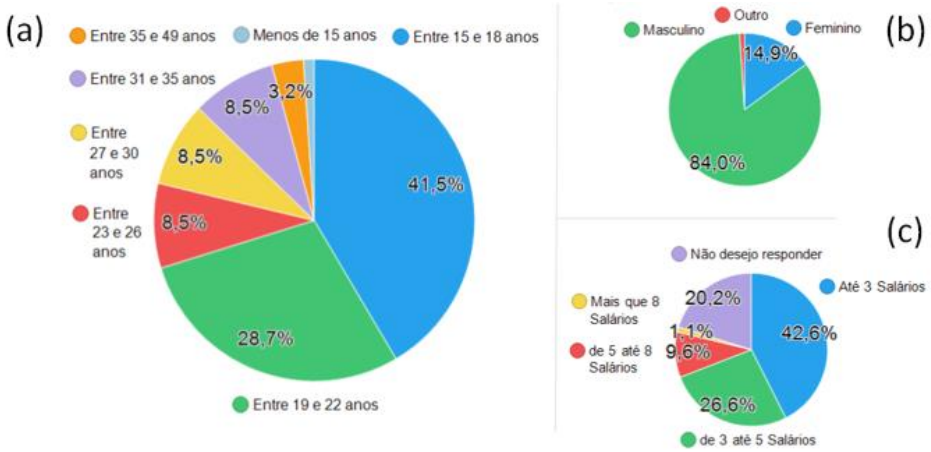


Figura 56 - (a) Idade; (b) Gênero; (c) Renda Familiar.

6.5.2. Questionário inicial da Competição

O questionário aplicado no início da competição contemplou objetivos como levantar algumas das características dos perfis dos alunos inscritos na competição Robocode Brasil 2016. A Tabela 24 apresenta o Questionário Inicial aplicado antes e durante as primeiras rodadas da competição ("idI" representa a identificação das questões iniciais, ex. "idI-1" refere-se a primeira questão do Questionário Inicial).

Tabela 24 - Questionário Inicial.

idI	Questão	Classificação	Objetivo
1	Como você classifica disciplinas que envolvem linguagem de programação de modo geral em relação ao grau de dificuldade?	Múltipla escolha – Avaliação	Pergunta opinião
2	Ainda com relação as disciplinas que envolvem linguagem de programação, selecione o(s) tipo(s) de aula(s) predominantemente utilizado(s) em sua Instituição?	Múltipla escolha – Avaliação	Pergunta ação
3	O que motivou sua participação na competição científica de programação Robocode Brasil?	Múltipla escolha – Mostruário (admite várias respostas)	Pergunta ação
4	De que modo se preparou para o torneio? Quais fontes utilizou para preparação?	Múltipla escolha – Mostruário (admite várias respostas)	Pergunta ação
5	Qual(quais) disciplina(s) que cursa ou cursou considera que auxiliou na preparação para torneio?	Múltipla escolha – Mostruário (admite várias respostas)	Pergunta opinião
6	Participou de outras iniciativas com envolvimento de jogos digitais dentro da sua Instituição? Se sim, qual (quais)?	Pergunta aberta / Perguntas de ação	Pergunta ação
7	Conhece outros jogos digitais que auxiliam no ensino de programação de computadores? Se sim, qual (quais)?	Pergunta aberta / Perguntas de ação	Pergunta ação

A questão “idI-1” exibe pontos relacionados as dificuldades de modo geral em cursar disciplinas de linguagem de programação. A Figura 57 exibe, como resultado, o grau de dificuldade predominantemente mediano (63,8%). Mais de 19% classificaram como difícil ou muito difícil e, aproximadamente, 17% enquadraram como fácil ou muito fácil. Um dado foi identificado neste cenário, caracterização de extremos nas respostas dentre 2 (duas) instituições participantes, IF Farroupilha e COTIL-UNICAMP em 2016. A instituição IF Farroupilha teve respostas gerais apontadas como médio (65,1%), difícil ou muito difícil (34,6%) e apenas 3,8% apontaram como fácil ou muito fácil. Por outro lado, a instituição COTIL-UNICAMP admitiu cerca de 42% fácil ou muito fácil e, aproximadamente, 57% como médio. Estas mesmas instituições apresentaram o seguinte resultado, em relação a colocação final na competição, na Liga LIAG do ano de 2015: IF Farroupilha foi campeã (2015) e uma das equipes do COTIL ficou na terceira colocação (2015).

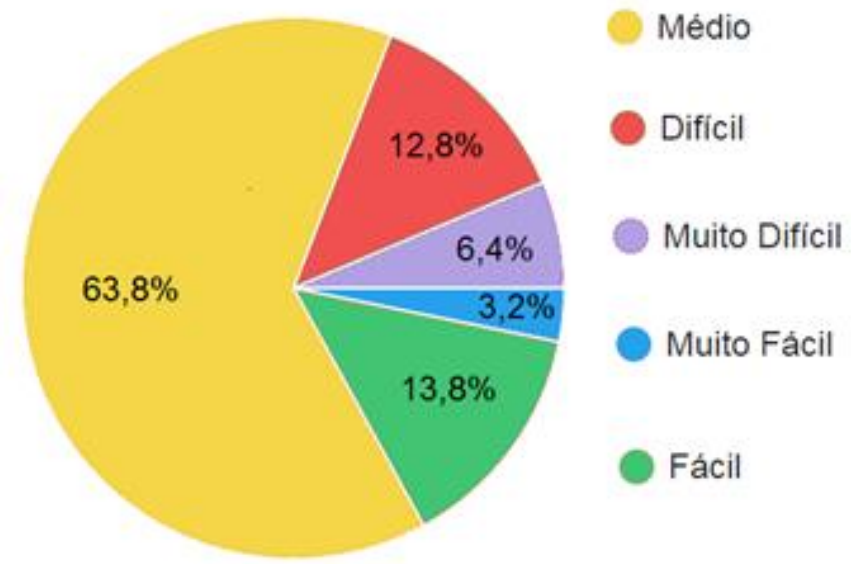


Figura 57 - Grau de Dificuldade na Disciplina de Programação.

Antes da resposta da segunda questão “idI-2” do Questionário Inicial, os seguintes tipos de aulas foram resumidamente explicados: (1) Aula expositiva clássica – mais comum em escolas de todos os níveis. Neste tipo de aula, o professor fala sobre um determinado assunto durante algum tempo, sendo que a postura dos alunos é predominantemente passiva; (2) Aula dialogada (dialógica) – o professor tenta quebrar a postura passiva dos seus alunos, por meio da introdução de questionamentos a serem respondidos pelos alunos, dinamizando a atividade em sala de aula; (3) Aula de demonstração – um tipo de aula em que o professor, com o uso de

equipamentos e outros materiais, inclusive experimentais, demonstra a operação, ou efeitos ou mesmo uma lei científica; (4) Aula prática – uso de equipamentos e materiais, com os quais os alunos fazem algum tipo de experiência sobre a lei científica ou efeitos dela, relacionado aos seus aspectos teóricos e práticos, sendo, por isso uma metodologia de trabalho ativa; (5) Aulas diversificadas – uso de problemas reais, jogos, grupos de pesquisa, debates, utilização de tecnologias atuais no auxílio do aprendizado. Alunos têm postura em buscar conhecimento mediados por professores.

Após a explicação de tipos de aulas, alunos responderam graus de incidência de cada tipo de aula na sua instituição. Para os tipos de aulas, as aulas clássicas receberam maior incidência da característica de "muito utilizada" (Figura 58). Aulas "clássicas" e "práticas" obedeceram uma sequência decrescente da característica "muito utilizada" para "nunca" com: (1) Altas incidências para "muito utilizada" e "utilizada"; (2) Baixas incidências para "pouco utilizada", "raramente" e "nunca". Aulas "práticas" concentraram características marcantes em um misto de "muito utilizada" e "utilizada". Uma das características da ementa da disciplina de linguagem de programação é a utilização de laboratórios para as práticas de programação. Neste contexto, esperavam-se incidências de utilizações para aulas práticas. Aulas "dialogadas" e aulas de "demonstração" receberam a característica predominante de "utilizada" e semelhanças nas demais incidências. Em contrapartida aos resultados anteriores, aula "diversificada" foi o único tipo que apresentou o resultado predominante "raramente" com algumas incidências de "nunca". O resultado deste gráfico mostra que iniciativas de aulas diversificadas ainda não estão populares quanto os demais tipos nas instituições participantes, pelo menos na área de informática. A competição Robocode foi planejada com características nas aulas diversificadas com objetivo de auxiliar disciplinas como linguagem de programação.

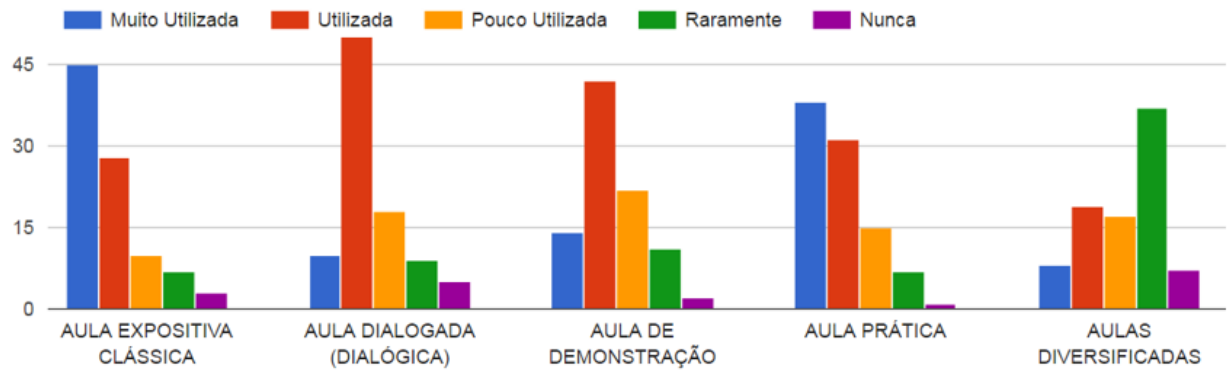


Figura 58 - Tipos de Aulas Predominantemente Utilizadas.

A pergunta “idI-3” teve por objetivo caracterizar as motivações que levaram os alunos a participar da competição. Essa foi a primeira questão que admitiu várias respostas igualmente válidas. Assim como observado na pesquisa de campo (subseção 6.3. Pesquisa de Campo Robocode Brasil), o questionário apresentou resultados similares quanto as motivações. A Figura 59 exibe as motivações principais, em porcentagens de incidência: (1) aprender – 59,6%; (2) competir – 49,5%; (3) jogar – 48,5% e (4) professores – 46,5%. As principais incidências das motivações da pesquisa de campo e da pergunta do questionário “idI-3” reforçam as palavras-chave da hipótese e do objetivo do presente trabalho: em relação a estimular a aprendizagem com jogos em competições científicas. A Figura 60 isola a média das 4 (quatro) principais incidências de motivações da pesquisa de campo e questionário. Mais uma vez, as principais motivações computadas nos questionários puderam reforçar os resultados apresentados na pesquisa de campo.

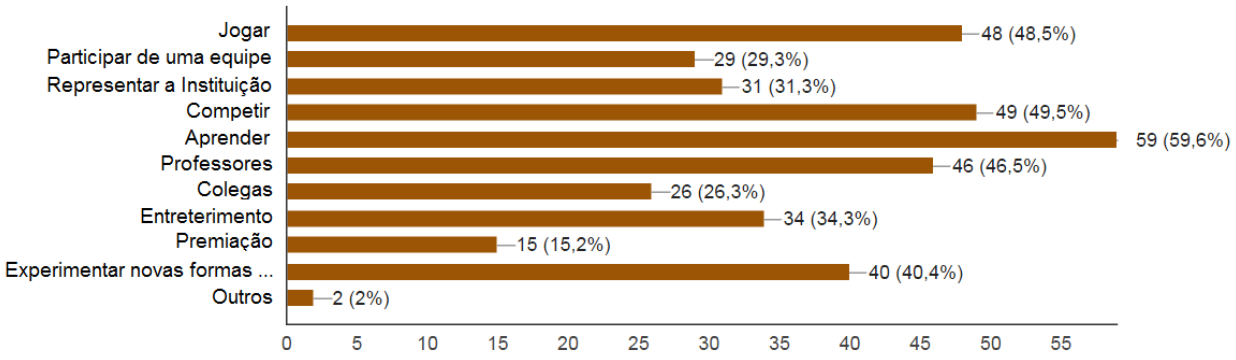


Figura 59 - Motivações para Participação no Robocode Brasil.

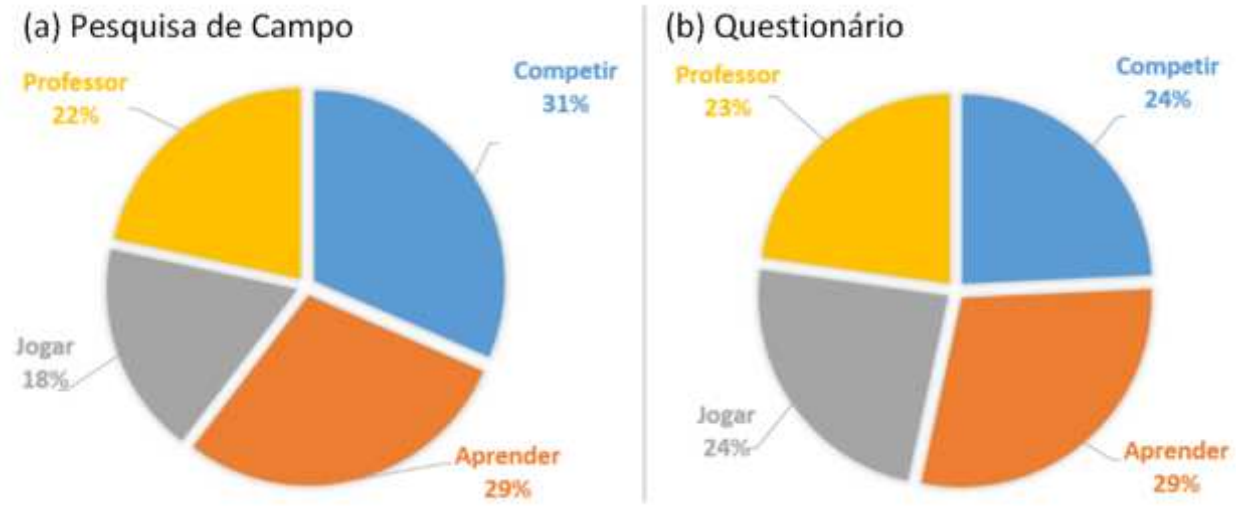


Figura 60 - Principais Motivações da (a) Pesquisa de Campo e (b) Questionário.

A questão “idI-4” teve por objetivo a identificação dos métodos de preparação adotados pelos alunos para disputar a competição 2016. A Figura 61 aponta que mais de 60% da preparação foi obtida com auxílio dos professores. A incidência do resultado, com apontamentos na figura do professor, pode ser relacionada com a fase de Orientações aos Gestores (subseção de seção 6.4.3. Orientações aos Gestores do Robocode Brasil). Pode indicar também que as equipes foram orientadas pelos gestores antes da competição e durante as primeiras rodadas. A segunda incidência de preparação remeteu a busca de materiais na internet sobre o ambiente Robocode. A terceira incidência indicou o estudo em casa; discussões neste item podem ser abertas no sentido em que os alunos estão buscando informações fora da competição, ou mesmo fora do ambiente escolar. A quarta incidência, mais uma vez, remete a uma das principais características do PBL no sentido propor soluções aos problemas mediante trabalho em equipe. Outras situações, como a baixa incidência de respostas a análise de torneios anteriores ou outros torneios no Brasil, pode indicar a ausência de uma referência brasileira do Robocode. Ao contrário de países como Irlanda, que detêm no portal oficial da competição denominado “GamesFleadh”, uma forte referência.

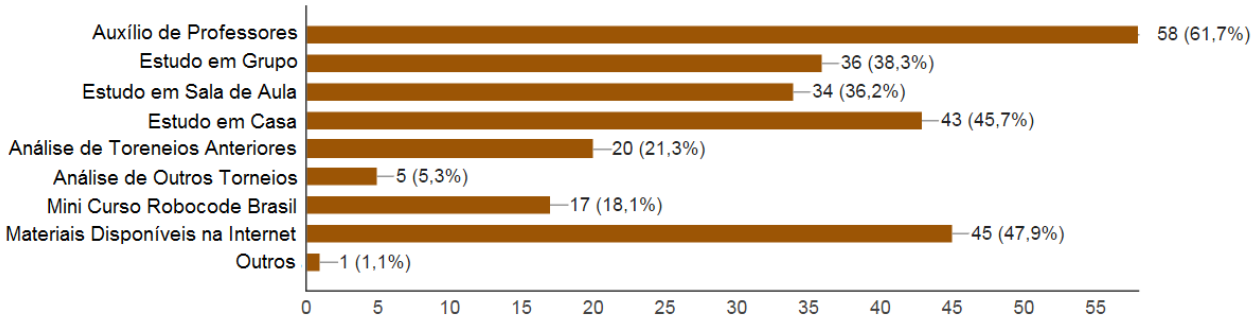


Figura 61 - Métodos de Preparação para Robocode Brasil.

A pergunta “idI-5” está relacionada a quais disciplinas auxiliaram os alunos durante o processo de preparação na competição (Figura 62). As disciplinas que trabalham diretamente com a programação de computadores foram mais indicadas. Mesmo que menos citadas, disciplinas como a Matemática e Inteligência Artificial (IA) podem gerar bons desafios no desenvolvimento do PBL. Disciplinas como Gestão de Projetos, que tratam tópicos como equipes, liderança, escopo, etc. não foram lembradas.

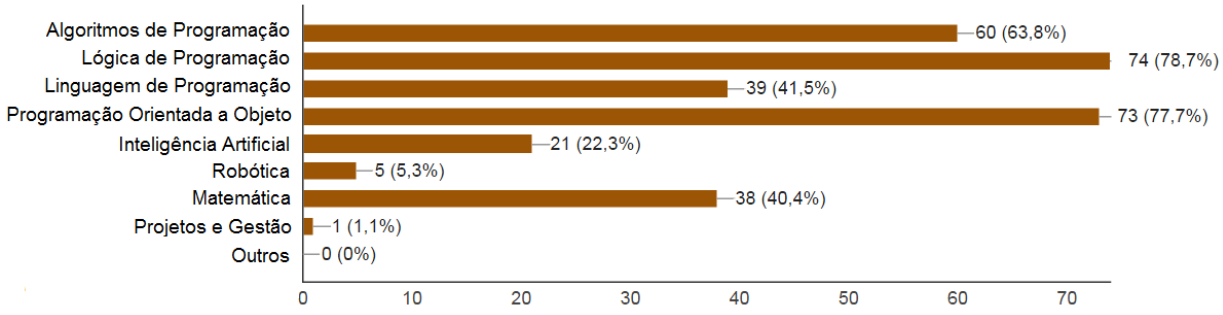


Figura 62 - Disciplinas que Auxiliaram na Preparação.

A "idI-6", apresentada na Figura 63, questionou sobre a participação dos alunos em outras iniciativas com envolvimento de jogos digitais dentro da instituição. Um alto índice, cerca de 80,9%, pôde ser observado de alunos que nunca participaram de nenhuma iniciativa com jogos dentro da sua instituição. Com pequena porcentagem (cerca de 5,3%), o evento internacional Global *Google Code Jam* foi lembrado. Competições de jogos não específicos diretamente ao ensino-aprendizagem de linguagem programação, como futebol e guerra (jogos para entretenimento) disputados por múltiplos usuários *on-line* em rede foram citados.

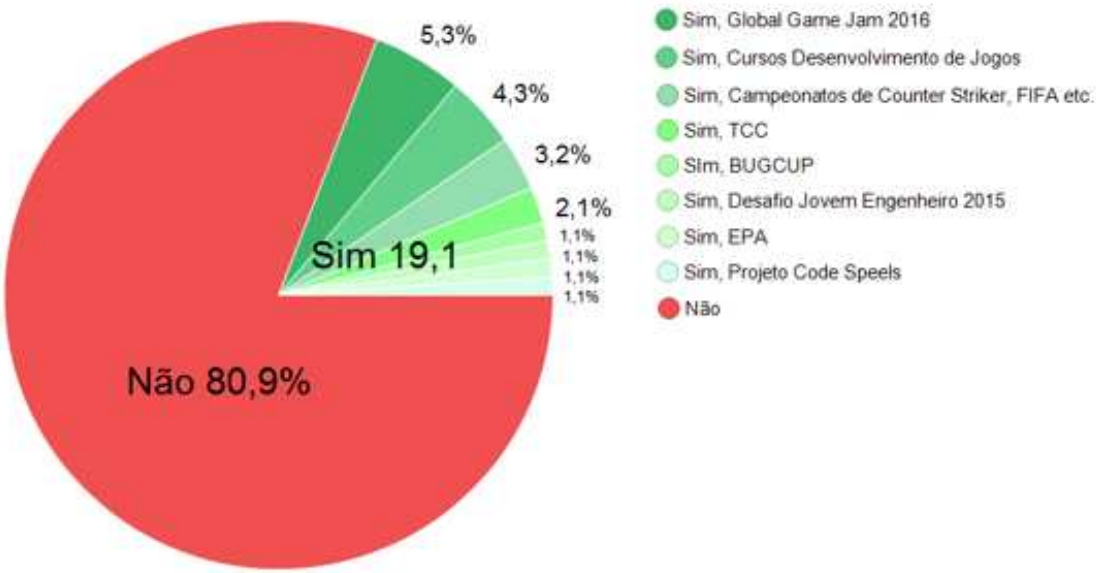


Figura 63 - Participação em Outras Iniciativas com Jogos.

A questão "idI-7" questionou os alunos acerca do conhecimento de jogos digitais específicos para ensino-aprendizagem de programação de computadores. O resultado mostrou-se alto, com cerca de 83% indicando não conhecer jogos digitais para ensino-aprendizagem de algum tipo de linguagem de programação (Figura 64). Aqueles que conhecem alguns,

apontaram ambientes como “Scracth” (5,3%) e “Minecraft” (2,1%) (detalhes destes ambientes na subseção de seção 3.2. Ambientes Educacionais no Apoio à Aprendizagem de Programação). Esse resultado reforça a questão “idI-6” e sua relação com “idI-7”, quando não existem participações ou iniciativas institucionais, então os alunos têm pouco (ou não têm) acesso ao conhecimento de jogos para aprendizagem de linguagem de programação.

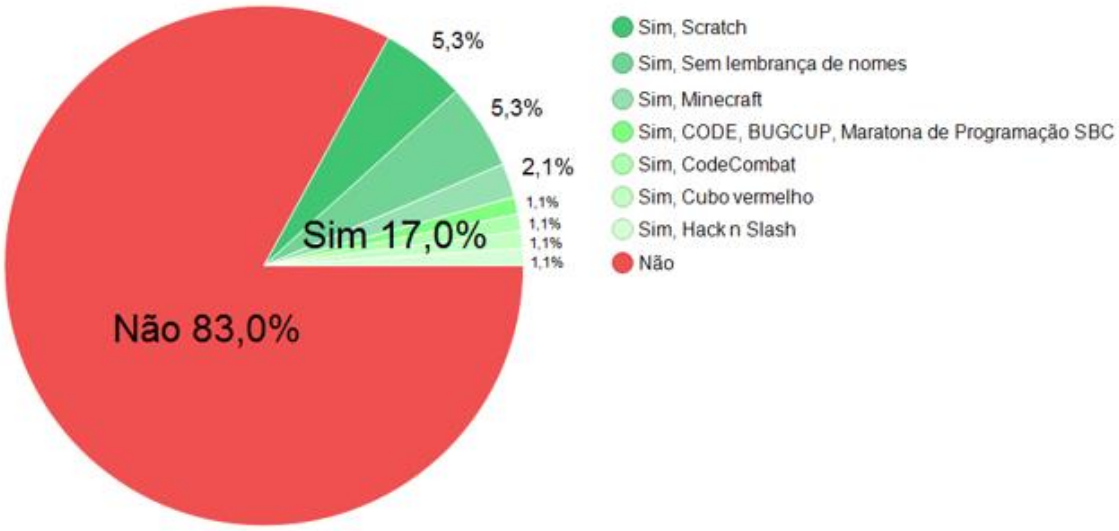


Figura 64 - Conhecimento de Jogos para Ensino de Programação.

6.5.3. Questionário final da Competição

Durante e ao final da última rodada de cada Torneio Local, o Questionário Final foi respondido por cerca de 75 alunos participantes do Robocode Brasil 2016. O Questionário Final teve por objetivo medir e avaliar características desenvolvidas ao longo dos Torneios. A avaliação das características foram delimitadas do seguinte modo: (1) Entre as metodologias usadas na competição (PBL, GBL e LBD); (2) Em comparações com metodologias adotadas nas aulas clássicas e as abordadas durante o Torneio; (3) Preferências de metodologias; (4) Existência de contribuições da competição Robocode; (5) Organização da competição; (6) Existência de evoluções na aprendizagem de conceitos de linguagem de programação; (7) Conclusões sobre a competição; (8) Avaliações de satisfações e motivações em participar da competição. A Tabela 25 apresenta o questionário final (“idF” representa a identificação das questões finais, ex. “idF-1” refere-se a primeira questão do Questionário Final).

Tabela 25 - Questionários Final.

idF	Questão	Classificação	Objetivo
1	Em uma Aula Expositiva Clássica (figura 1) de 60 minutos da disciplina de linguagem de programação, quanto tempo você considera que consegue prestar atenção? * Aula de 60 minutos com apresentação de conceitos como: sintaxe; variáveis; estruturas condicionais; estrutura de repetição.	Múltipla escolha – Mostruário (admite uma resposta)	Pergunta ação
2	Em uma Aula Diversificada com envolvimento de equipes e jogos digitais (figura 2) de 60 minutos da disciplina de linguagem de programação, quanto tempo você considera que consegue prestar atenção? * Aula de 60 minutos com apresentação de conceitos como: sintaxe; variáveis; estruturas condicionais; estrutura de repetição.	Múltipla escolha – Mostruário (admite uma resposta)	Pergunta ação
3	Entre: Expositiva Clássica (figura 1) e Aula Diversificada com Jogos (figura 2), qual a sua preferência se pudesse escolher seus tipos de aulas.	Múltipla escolha – Mostruário (admite uma resposta)	Pergunta ação
4	Considera que o torneio Robocode auxiliou e/ou melhorou o entendimento de conceitos e práticas de linguagem de programação?	Pergunta fechada	Pergunta opinião
5	Considera que os jogos digitais (ex. Robocode e outros) poderiam ser ferramentas utilizadas nas aulas presenciais para auxílio na compreensão de conceitos e práticas?	Pergunta fechada	Pergunta opinião
6	Desafios como melhorar a movimentação do Robô (ex. método run() ou método onHitWall()) são encontrados durante a competição. Em uma escala de 0 a 5 quanto você acha que evoluiu os métodos de movimentação do seu Robô da primeira à última etapa disputada?	Múltipla escolha – Avaliação	Pergunta opinião
7	Quais características você e sua equipe adotaram para resolver os desafios encontrados no jogo durante a competição?	Múltipla escolha – Mostruário (admite várias respostas)	Pergunta ação
8	Se o Robocode auxiliou e/ou melhorou a compreensão, quais conceitos considera que houve melhoria?	Múltipla escolha – Mostruário (admite várias respostas)	Pergunta ação
9	SE pudesse escolher, gostaria de estudar conceitos e práticas de programação em aulas clássicas OU em diversificação de metodologias que envolvessem o ensino-aprendizagem com base em jogos?	Múltipla escolha – Mostruário (admite uma resposta) / Pergunta Ação	Pergunta ação
10	Considera motivador programar em ferramentas como o Robocode dentro de competições? Se sim, o que ela pode trazer de diferencial em relação ao ensino tradicional (aula clássica)?	Múltipla escolha – Mostruário (admite várias respostas)	Pergunta ação
11	Avaliar a Motivação Satisfação com Torneio. 	Pergunta aberta	Pergunta ação
12	Sugestões	Pergunta aberta	Pergunta opinião

As questões “idF-1” e “idF-2” admitem cenários para apresentação de conceitos introdutórios de linguagem de programação (sintaxe, variáveis, estruturas condicionais e estruturas de repetição) em aulas de 60 minutos. Foram questionados quanto tempo em minutos os participantes conseguem prestar atenção: (1) Na questão “idF-1” em aulas clássicas; (2) Na questão “idF02” em aulas diversificadas com envolvimento de equipes e jogos digitais. Foi

possível comparar os resultados entre as questões “idF-01” e “idF02” na Figura 65. O resultado apresentado na questão “idF-1” (aula clássica), Figura 65 (a), foi apenas 4,4% enquanto na “idF-2” (aula diversificada), Figura 65 (b), cerca de 51,5% dos alunos admitiram conseguir prestar atenção entre 51 a 60 minutos. No quesito prestar atenção entre 41 a 60 minutos foi possível comparar o resultado de 32% no “idF-1”, Figura 65 (a), contra 85,3% apresentado no “idF-2”, Figura 65 (b). Os alunos participantes dos torneios admitiram, preferencialmente, dispor maiores fatias de tempo em prestar atenção em conceitos apresentados em aulas diversificadas.

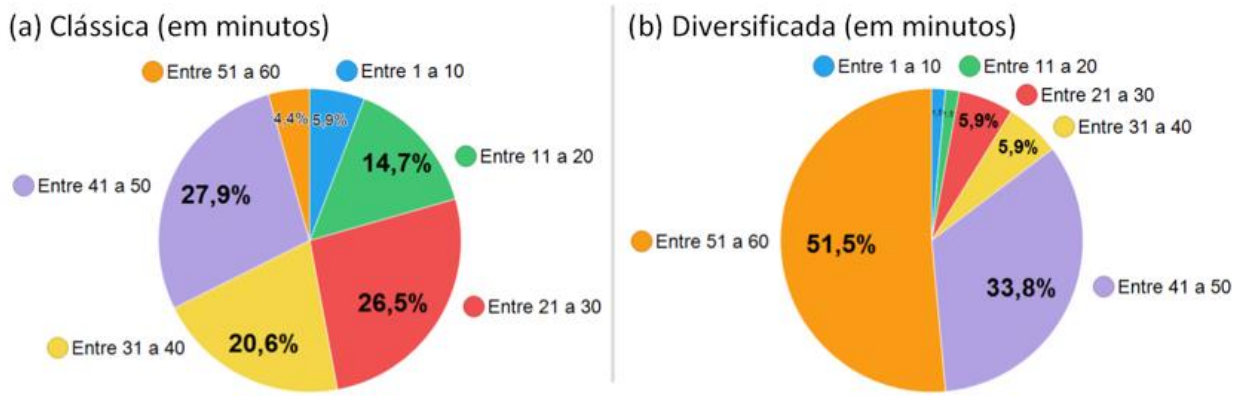


Figura 65 - (a) Aula Clássica (b) Aula Diversificada.

Apoiado pelos conceitos tratados nas questões “idF-1” e “idF2”, a questão “idF-3” foi relacionada à preferência quanto ao tipo de aula. A Figura 66 (a) auxilia o aluno a identificar o cenário da aula clássica enquanto a Figura 66 (b) o cenário da aula diversificada. O tipo de questão adotado para “idF-3” foi “múltipla escolha” com objetivo de “pergunta ação” que admitiu as seguintes opções: (1) Expositiva clássica (figura 1); (2) Predominantemente expositiva clássica (figura 1); (3) Misto entre expositiva clássica (figura 1) e aula diversificada (figura 2); (4) Predominantemente aula diversificada (figura 2); (5) Aula diversificada (figura 2). A opção apenas “expositiva clássica (figura 1)” não recebeu nenhuma incidência de resposta, portanto a mesma não pôde ser computada nos resultados da Figura 66 (c). Aula predominantemente clássica surgiu com somente 1,5% da preferência. A maior porcentagem de resposta remeteu à preferência de aulas mistas entre clássicas e diversificadas com 55,9%. Porém a preferência por aula diversifica ou predominantemente diversificada foi amplamente selecionada com cerca de 42,6%. Este resultado indicou que os alunos participantes da competição Robocode Brasil 2016 estão buscando alternativas as aulas tradicionais.

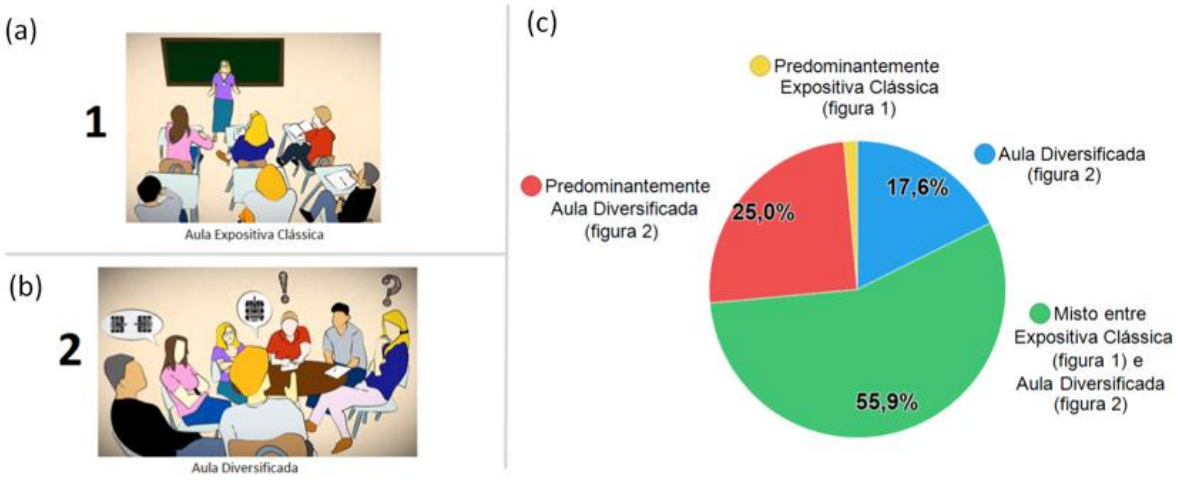


Figura 66 - (a) Aula Clássica (b) Aula Diversificada (c) Preferência de Aula.

A questão "idF-4" perguntou aos alunos se a competição Robocode auxiliou e/ou melhorou o entendimento de conceitos e práticas presentes na ementa da disciplina de linguagem de programação. A Figura 67 (a) exibe cerca de 89,7% do resultado "sim", reconhecendo avanços nos conceitos com auxílio da competição. A resposta "não" apresentou o resultado de 8,8% do total enquanto a opção outros com 1,5% foi respondida "em partes". Em complemento a "idF-4", a questão "idF-5" perguntou se os jogos digitais educacionais, com ênfase no ambiente Robocode, poderiam ser adotadas como ferramentas em aulas presenciais para auxílio na compreensão de conceitos e práticas de linguagem de programação. O resultado apresentado na Figura 67 (b) apontou, mais uma vez, uma ampla margem para resposta "sim" (97,1%) para adoção de propostas como Robocode em aulas de programação.

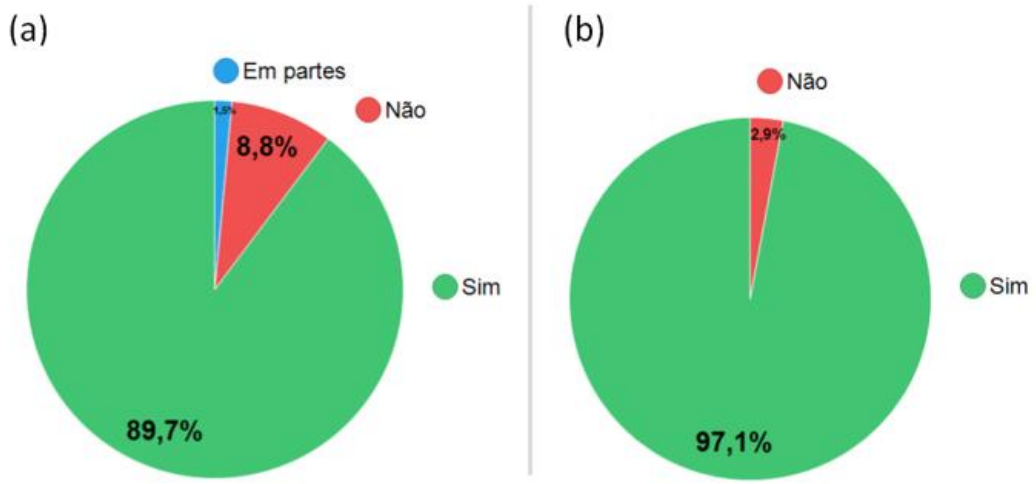


Figura 67 – Auxílio no Aprendizado (a) Torneios (b) Jogos com ênfase Robocode.

A questão “idF-6” foi diretamente relacionada aos problemas (desafios) propostos de movimentação do robô ao longo das disputas das rodadas da competição. A questão “idF-6” perguntou aos alunos em uma escala de 0 a 5, o quanto houve evolução na programação dos métodos e eventos de movimentação do robô desde a primeira até a última rodada disputada. Consistiu em uma questão avaliativa de opinião. A Figura 68 destaca o resultado para média evolução com cerca 44,1% com nível de evolução “3”. Níveis com pouca (“2”) ou nenhuma evolução (“1”) ficaram cerca de 17,6%. Níveis mais elevados de evolução (“4” e “5”) registraram cerca de 38,3%.

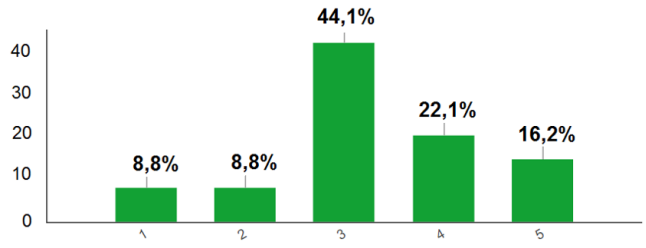


Figura 68 - Evolução da Programação do Robô.

Para a questão “idF-7” fez-se importante ressaltar que os processos da metodologia PBL admitem etapas e processos distintos para determinação dos problemas. Os processos PBL foram utilizados para determinar os problemas gerados automaticamente durante as disputas na competição Robocode Brasil 2016. A Tabela 26 exibe os principais processos do PBL e sua abordagem dentro da competição Robocode Brasil.

Tabela 26 – Processos do PBL Associados à Competição

Etapa	Processos Metodologia PBL	Abordagem PBL na Competição
1	1. Identificação e delimitação do problema	1. Analisar os problemas e as estratégias suportadas na competição e no jogo.
	2. Formular os objetivos de aprendizagem a partir da discussão do problema.	2. Formular objetivos de acordo com os métodos do robô (ex. ajustar a movimentação do robô).
2	3. Levantar as hipóteses possíveis para resolução do problema	3. Promover debates e discussões na equipe para resolver os problemas.
	4. Rediscussão do problema com base nos novos conhecimentos adquiridos	4. Rediscutir os problemas com base nos novos conceitos de linguagem de programação e estratégias aprendidos.

A questão “idF-7” teve por objetivo principal analisar se os 4 (quatro) principais processos do PBL foram atendidos durante a resolução dos problemas propostos na abordagem da competição. A Figura 69 apresentou os resultados, nos quais é possível separar os termos “sempre” e “frequentemente”, amplamente citados e reiterados na ordem dos processos e

abordagens da Tabela 26: (1) 65,4% – identificação e estratégias; (2) 72,3% – formulação de objetivos; (3) 58,3% – discussões em grupo; (4) – 68,1% rediscussão dos problemas. Para a utilização de modo "moderado" o resultado estabeleceu-se: (1) 27,8% – identificação e estratégias; (2) 19,4% – formulação de objetivos; (3) 26,4% – discussões em grupo; (4) – 22,3% rediscussão dos problemas. Para "raramente" ou "nunca" houve baixos níveis de incidências: (1) 6,8% – identificação e estratégias; (2) 8,3% – formulação de objetivos; (3) 15,3% – discussões em grupo; (4) – 9,6% rediscussão dos problemas.

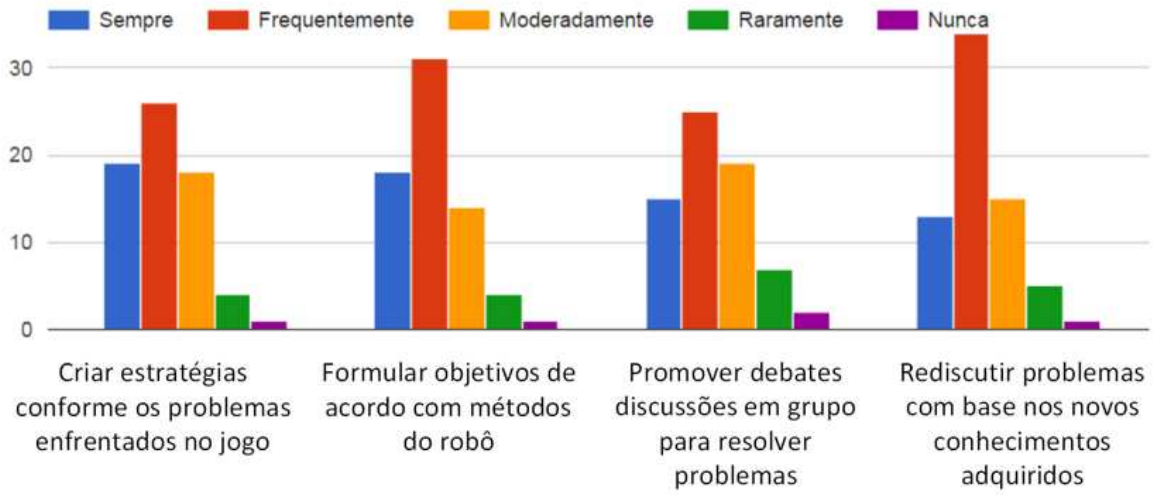


Figura 69 - Utilização dos Processos PBL na Competição.

A questão "idF-8" procurou investigar se o ambiente Robocode auxiliou e/ou melhorou a compreensão de conceitos introdutórios de linguagem de programação. A questão admitiu múltiplas respostas e seu resultado é apresentado na Figura 70. O destaque pôde ser observado na melhoria de aspectos de conceitos relacionados a sintaxe (64,7%), seguido de estruturas de condicionais (54,5%), repetição (51,5%) e variáveis (23,5%). Uma informação relevante relacionou-se a baixa incidência daqueles que responderam que Robocode não auxiliou e/ou não houve nenhuma melhoria na aprendizagem dos conceitos, com apenas 5,9%.

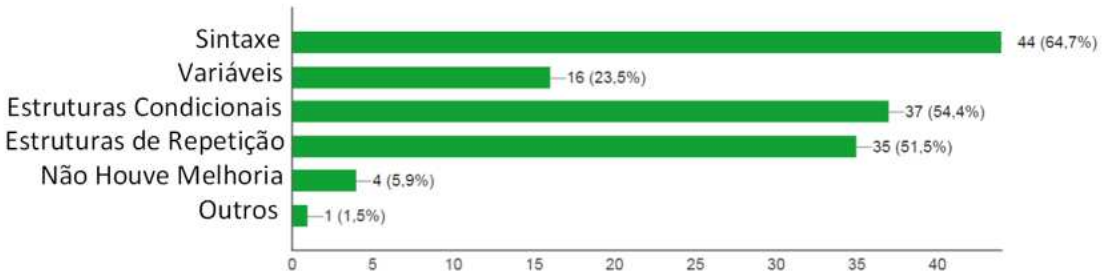


Figura 70 - Melhoria do Entendimento dos Conceitos.

A questão “idF-9”, com objetivo de ação, solicitou ao aluno para eleger o tipo de aula no qual gostaria de estudar conceitos introdutórios e práticas de programação. As opções apresentadas de múltipla escolha com admissão de única resposta verdadeira foram: (1) aulas “clássicas”; (2) aulas “clássicas” e aulas “diversificadas com jogos”; (3) aulas diversificadas com jogos; (4) outros tipos (em caso de seleção foi possível informar outros tipos). Os resultados podem ser observados na Figura 71. Aulas “clássicas” e outros tipos não tiveram nenhuma incidência. Aulas “clássicas” juntamente com “diversificadas com jogos” apresentaram o resultado de 70,6% e aulas apenas “diversificadas com jogos” resultou em 29,4% das escolhas de preferências.

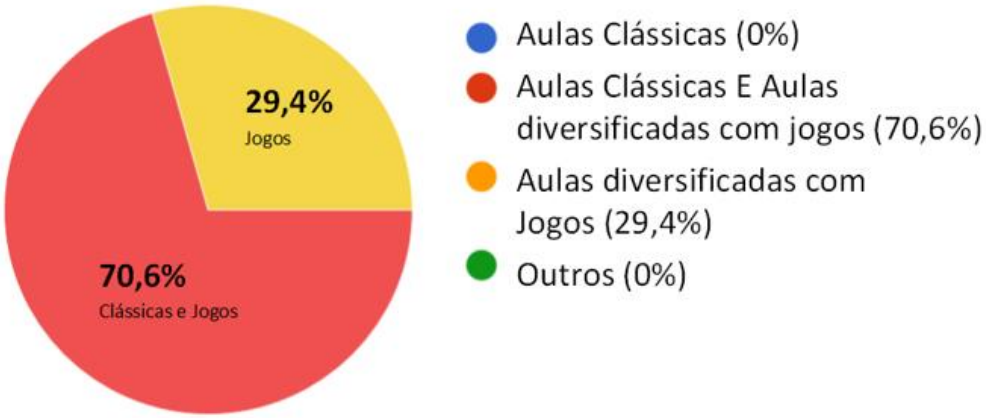


Figura 71 - Preferência de Tipos de Aulas.

As múltiplas escolhas da décima questão, “idF-10”, foram desenvolvidas com base em questões aplicadas na Liga LIAG 2015. As questões aplicadas em 2015 fizeram referências com objetivo de buscar motivações das participações dos alunos nos torneios e como estes eventos puderam auxiliar no aprendizado de linguagem de programação (Meira et al., 2016). Foi utilizado o Diagrama de Afinidades (DA) para desenvolver grupos comuns de respostas. O DA teve como objetivo agrupar ideias, opiniões e informações conforme afinidade que apesentaram entre si. Para idealização do DA, foram obtidas respostas abertas no ano de 2015 dos participantes da competição. As respostas escritas ou verbais foram agrupadas em informações de acordo com afinidade apresentada entre si. Cada agrupamento de informações recebeu um rótulo que descreveu de forma objetiva as afinidades. Esses agrupamentos formaram as múltiplas escolhas da questão “idF-10” observada na Figura 72.

A décima questão, “idF-10”, perguntou aos alunos se consideraram motivador programar em ferramentas como o Robocode dentro das competições científicas. Em caso de

resposta positiva, o aluno teve acesso as múltiplas escolhas, com admissão de várias respostas para a pergunta ação. Seu resultado pode ser observado de acordo com a porcentagem de incidências de respostas na Figura 72. A maior incidência de respostas, com cerca de 75,3%, admitiu a motivação na participação de um ambiente divertido para desenvolver as competências presentes nas ementas das disciplinas de linguagem de programação. Seguida pela incidência de 60,3% com compromisso no torneio em acompanhar os resultados possibilitou criar um senso crítico na tentativa de constante melhoria dos programas e da estratégia de jogo. Motivações com cerca de 50% das incidências foram obtidas: (1) 54,8% - Trabalhar em equipes para desenvolver estratégias para o Robô; (2) 53,4% - competição estimula o interesse em aperfeiçoar o código com objetivo de vencer; (3) 52,1% - despertar a curiosidade intelectual; (4) 52,1% - situações encontradas no jogo encorajaram a pesquisar de forma criativa e exploratória; (5) 50,7% - estudou conceitos ainda não vistos nas aulas; (6) 50,7% - melhorias no processo de aprendizagem; (7) 47,9% - geração de interações entre as equipes participantes, colegas e professores. Outras incidências com cerca de 40% foram selecionadas: (1) 43,7% - aprofundou a pesquisa de conceitos abordados em disciplinas de linguagem de programação; (2) 42,5% - ferramenta complementar ao estudo de linguagem de programação; (3) 39,7% - conferiu o os resultados e a evolução durante a competição; (4) 39,7% - auxiliou o entendimento de conceitos e práticas.

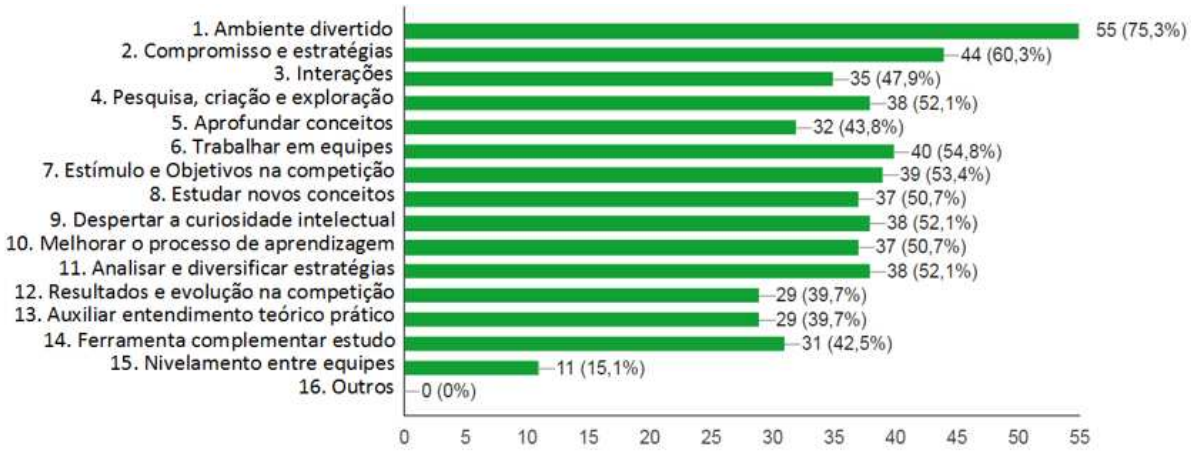


Figura 72 - Motivações com Robocode e Competições Científicas.

A pergunta “idF-11” foi relacionada a avaliação de satisfação. Para essa avaliação foi utilizada estratégia com base na seleção de pictogramas, amplamente utilizada em redes sociais para expressar opiniões. Pictograma transmite a ideia de uma palavra ou frase, popularmente conhecidos como “emoji” de origem japonesa, composta pela junção de elementos (imagem e

letra). A estratégia adotada para aferir o grau de satisfação para representações entre “muito satisfeito” (☺), e “muito insatisfeito” (☹), conforme observado na Figura 73. O resultado da Figura 73 apresentou o grau de satisfação para os seguintes itens da competição: (1) satisfação geral do Robocode Brasil 2016; (2) satisfação nas instituições onde foram realizados os torneios; (3) satisfação com a equipe; (4) resultados da equipe; (5) satisfação com envolvimento da instituição.

A satisfação geral com Robocode Brasil 2016 apresentou-se do seguinte modo: (1) 33,4% – muito satisfeito; (2) 48,0% – satisfeito; (3) 10,6% – indiferente; (4) 4,0% – insatisfeito; (5) 4,0% – muito insatisfeito. A satisfação dos torneios locais institucionais e torneio público resultou em: (1) 28,0% – muito satisfeito; (2) 46,7% – satisfeito; (3) 14,7% – indiferente; (4) 9,3% – insatisfeito; (5) 1,3% – muito insatisfeito. A satisfação dentro das equipes apresentou: (1) 34,7% – muito satisfeito; (2) 45,3% – satisfeito; (3) 8,0% – indiferente; (4) 5,3% – insatisfeito; (5) 6,7% – muito insatisfeito. A satisfação com os resultados conquistados pela equipe dentro da competição: (1) 22,7% – muito satisfeito; (2) 49,3% – satisfeito; (3) 12,0% – indiferente; (4) 9,3% – insatisfeito; (5) 6,7% – muito insatisfeito. A última variável mediu a satisfação com envolvimentos institucionais na competição Robocode Brasil 2016 com: (1) 25,4% – muito satisfeito; (2) 40,0% – satisfeito; (3) 24,0% – indiferente; (4) 5,3% – insatisfeito; (5) 5,3% – muito insatisfeito (Figura 73).

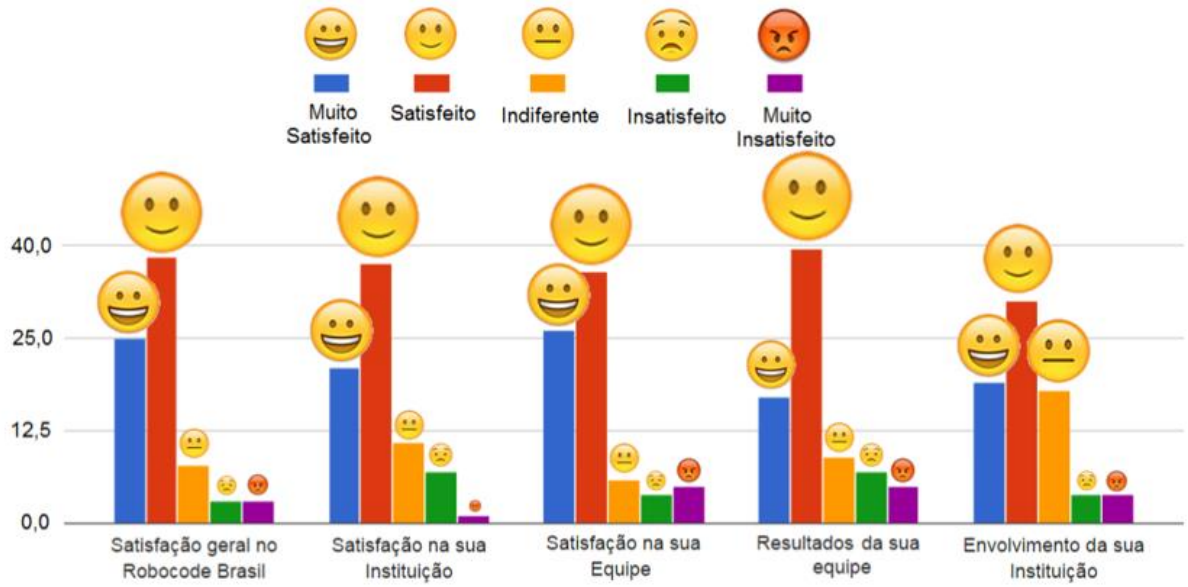


Figura 73 - Grau de Satisfação com Robocode Brasil 2016.

A medição da variável de envolvimento das instituições indicou que ainda existem ações a serem discutidas entre Administradores da Competição e Gestores Institucionais com objetivo em ampliar promoção e apoio local ao evento Robocode Brasil. Em contrapartida, as demais variáveis indicaram ótimos resultados, com altas incidências de satisfação. Esses resultados puderam comprovar que a partir do planejamento associado com gestores dos torneios institucionais, foi possível desenvolver uma competição científica de satisfação geral positiva.

7. CONCLUSÃO

Alunos da geração de nativos digitais inseridos em um contexto de conectividade no universo digital tendem a considerar ambientes de ensino tradicionais desmotivadores. Essa desmotivação pode interferir na estruturação do conteúdo aprendido, enfatizando um conhecimento apenas superficial. Este trabalho propôs uma nova abordagem para ensino-aprendizagem, com geração de interações, apropriando-se de ferramentas presentes no cotidiano dos alunos para ensino de linguagem de programação.

Um ambiente de competição científica de programação trabalha com diversas situações problemas que permitem desenvolver a colaboração. A estruturação da metodologia PBL na competição científica encoraja não somente um conhecimento mais profundo de uma determinada linguagem de programação, Java, mas também de estruturas lógicas que de fato resolvem um problema. O jogo educacional Robocode pode despertar o interesse no aprendizado de conceitos de linguagem de programação a partir de elementos que facilitam o uso de determinadas estruturas lógicas.

Diferentes abordagens foram utilizadas para desenvolvimento do presente trabalho. A metodologia *Problem Based Learning* (PBL) foi usada nas gerações das situações-problemas no Robocode e *Learning by Design* (LBD) foi usada para práticas pedagógicas no sequenciamento estrutural do problema. As técnicas de pesquisa adotadas foram: pesquisa-ação no planejamento do Robocode Brasil 2016 com base nos problemas coletivos enfrentados na Liga LIAG 2015; pesquisa de campo em duas instituições para coletar informações da experiência nos Torneios Locais (2016); entrevistas realizadas para desenvolvimento colaborativo do edital da competição (2016); aplicação de questionários para avaliar a experiência dos alunos junto aos torneios (2016). E, por fim, o ambiente Robocode, dividido em: *framework*, ambiente para programação do jogo (*design* e estratégias do robô); e o ambiente de jogo, local das disputas e análises dos robôs.

A evolução da competição ocorreu com base nas experiências coletivas vivenciadas em uma liga similar conduzida em 2015. O Robocode Brasil 2016 nasceu com uma gestão de competição descentralizada e estruturada em condições para receber uma quantidade maior de instituições e alunos. A Administração da Competição desenvolveu um edital. Neste, uma nova proposta para a condução foi sugerida para 2016, incluindo o papel do professor Gestor institucional aplicado a cada Torneio. Além da gestão, os professores gestores tinham como

meta instigar as equipes na constante evolução dos robôs, conforme análises de jogos, para definir e estruturar problemas com base na PBL e LBD. Os questionários respondidos pelos alunos competidores apontaram os Gestores como principais motivadores e apoiadores no treinamento das equipes, o que indica o acerto desta proposição. Essas características também foram evidenciadas durante a competição: nos instantes em que ocorriam as evoluções estratégicas e da programação de novos conceitos da linguagem, surgiam novas motivações com novos desafios, em geral, orientados por gestores para avançar na competição.

Competir, aprender, jogar e a orientação dos professores foram principais motivações dos alunos apontadas pelas técnicas de pesquisas de campo e questionários. As motivações puderam ser relacionadas com o objetivo geral do trabalho: em estimular a aprendizagem com jogos em competições científicas. A apresentação dos processos PBL pelos Gestores e sua ampla utilização pelas equipes colaborativas, durante a competição, ficou registrada nos resultados da motivação para objetivos específicos do trabalho.

Alunos participantes da competição foram caracterizados predominantemente pelo gênero masculino. Embora a maioria das turmas fosse predominantemente masculina, haviam exceções nos cursos técnicos integrados em informática, com turmas equilibradas em gêneros. Contudo, mesmo em turmas mistas, prevaleceram as participações do gênero masculino. Também foi identificado que grande parte dos participantes tinham uma idade entre 15 a 22 anos e baixa renda familiar, mostrando que a competição pôde apoiar à aprendizagem de jovens de baixa renda.

Nas instituições participantes, aulas “clássicas” e “práticas” caracterizaram como os principais tipos de aulas frequentemente utilizadas, enquanto as aulas “diversificadas” ainda não apresentaram resultados de boa aderência. A maioria dos alunos apontou suas preferências nas aulas concomitantes “clássicas” e “diversificadas com jogos”. Outros sinalizaram a preferência em estudar exclusivamente em aulas “diversificadas com jogos”. Aulas unicamente “clássicas” não foram lembradas. A diversificação de tipos de aulas, contemplando envolvimento de jogos e equipes, têm maiores chances de atrair atenção e estimular estudos de conceitos de programação.

Nos questionários, alunos responderam não conseguir prender atenções durante o tempo total de uma aula “clássica”. Em contrapartida, mais da metade dos alunos considerou prender suas atenções na totalidade ao participar de aulas “diversificadas” com envolvimento de equipes

e jogos. Alunos indicaram que o ambiente da competição é favorável para dedicação de maior fatia de tempo na aprendizagem dos conceitos de programação. Grande parte dos alunos nunca tinha tido uma aula "diversificada", e mesmo em primeira experiência, indicou a preferência em relação à aula "clássica", o que mostra que os métodos utilizados no Robocode Brasil 2016 podem ser abordagens interessantes na aprendizagem de programação.

Dentro das instituições participantes, 80,9% dos alunos nunca participaram de outras iniciativas com jogos, tendo sido a primeira experiência da maioria. Entrevistas com gestores revelaram a baixa incidência de iniciativas. Para *Serious Games* de programação, 83% desconheceram qualquer jogo ou ambiente com essa finalidade; houve poucas citações de "Scratch" ou "Minecraft", e outros 5,3% não recordaram nomes. Ficou evidenciado que pode ser interessante que as instituições desenvolvam iniciativas nas temáticas dos jogos digitais na área de informática. Para o Robocode Brasil 2016, 89,7% dos alunos admitiram auxílio da competição no entendimento de conceitos. Outros 97,1% consideraram os jogos, ênfase no Robocode, como ferramentas de auxílio a serem utilizadas nas aulas. Os resultados apontaram o interesse dos alunos no ensino-aprendizagem com novas propostas diversificadas com *Serious Games*.

O aprendizado foi percebido em consequência das motivações nas resoluções dos desafios a cada rodada. Estes desafios foram percebidos na competição, com estímulos do interesse das equipes no constante aperfeiçoamento do robô, para vencer ou atingir pontuações de classificações do jogo. Para as disputas, existiu evolução da programação dos robôs. Quando estimulados, tendências de evolução gradativa dos conceitos de programação puderam ser compreendidos com auxílio do jogo. O planejamento do ambiente da competição associado com as interações dos professores e equipes potencializaram resultados com situações de jogo que encorajam pesquisas criativas e exploratórias com PBL. A curiosidade intelectual frente a diferentes soluções possíveis no jogo, foram representadas nas conquistas, aperfeiçoamento ou mudanças de estratégias de programações dos atributos do robô.

A proposta do edital com ciclo de abertura dos códigos fonte dos robôs durante o torneio favoreceu a troca de experiências entre equipes, como movimentação do robô, com base nos conceitos de "comandos de decisões" ou "estruturas de repetição" discutidos no presente trabalho. Existiram as discussões de estratégias e experiências de jogo compartilhadas entre equipes. A geração de interações entre as equipes, colegas e professores foi uma das motivações determinantes ao aprendizado. Trabalhar com interação na programação permitiu às equipes

desenvolverem habilidades colaborativas para resolução dos desafios. As hipóteses colaborativas levantadas proporcionaram discussões em grupo, a fim de eleger as melhores soluções para os desafios. O compromisso em acompanhar os resultados dos torneios possibilitou desenvolver senso crítico na tentativa de constante melhoria dos programas e da estratégia de jogo.

Com base no que foi observado, pode-se dizer que há indícios que torneios com envolvimento de jogos digitais, como Robocode, podem auxiliar na diversificação da aprendizagem. A associação entre: planejamento colaborativo sistemático da competição; estudo e aplicação do ambiente educacional Robocode; metodologias PBL e LBD; e o envolvimento dos professores das instituições promoveram um ambiente lúdico e motivador para desenvolver as competências presentes nas disciplinas de linguagem de programação. Os ambientes dos Torneios geram situações-problemas que podem incentivar alunos a buscar soluções, na aprendizagem de novos conceitos, como aperfeiçoamento de estratégias de movimentação dos robôs. Equipes formadas por alunos de diferentes níveis de conhecimento, tiveram a oportunidade de estabelecer discussões acerca dos conceitos aprendidos da linguagem. Resultados positivos, enfatizados pelas instituições, entrevistados e alunos, indicaram o Robocode associado aos Torneios como um ambiente interessante para auxiliar no estímulo do processo de aprendizagem de linguagem de programação de modo lúdico.

Ao final, pode-se observar que o desenvolvimento de métodos diversificados em competições científicas com jogo educacional Robocode pode ser uma abordagem efetiva quando o objetivo é auxiliar a estimular alunos na aprendizagem de conceitos de linguagem de programação. O trabalho realizado com Robocode Brasil 2016 concluiu na satisfação geral com cerca de 75% dos alunos, participantes dos questionários, estiveram satisfeitos ou muito satisfeitos com os resultados da competição. Essa alta incidência de porcentagem de satisfação, também, pode ser caracterizada como estímulo ao aprendizado. O Robocode Brasil 2016 serviu como ação complementar em disciplinas de programação nos cursos da área de informática.

7.1. Trabalhos Futuros

Uma motivação pouco citada como preparação para o Torneio foi a análise de torneios anteriores ou outros torneios no país. A proposta do Robocode Brasil 2016 teve em sua missão

promover a maior competição brasileira e com isso posicionar-se como nome de referência em Robocode. Futuramente, o Robocode Brasil poderá trabalhar como um repositório oficial livre do Robocode. Construir um ambiente-referência para disponibilização de consultas e análises de códigos, históricos, versões, explicações das técnicas utilizadas e robôs em seu estado da arte para aprendizagem e treinamentos em competições futuras pode ser interessante, podendo cadastrar Torneios e servir de base de dados de editais e regras de competições Robocode.

Os trabalhos futuros remetem a expansão da competição Robocode Brasil, incorporando torneios similares ao Robocode com simulações de disputas entre navios virtuais no Naval Robocode. Essa nova modalidade permitirá ganhos na dinâmica do jogo a partir dos conceitos aprendidos no Robocode. Com o Naval Robocode novos modos de programações e estratégias estarão presentes durante os torneios, como definições de múltiplos disparos ou identificação de obstáculos postados no oceano do jogo. Essas características do Naval Robocode poderão ser estruturadas durante os torneios com objetivo em desenvolver novas possibilidades de jogo para estimular a aprendizagem programação.

Para plataforma Robocode Brasil, propostas futuras para desenvolver a evolução do seu estado atual, de concentradora de códigos e com interferências dos gestores e administradores para registro dos resultados das disputas, para expansão das funcionalidades em versões automatizadas. Essa evolução consiste em desenvolver uma plataforma *on-line* programável com recursos para após inscrição dos códigos dos robôs de equipes, capacidade de automatizar as disputas no ambiente Robocode, recuperar as informações do jogo e devolver os resultados.

Não houveram questionários aos alunos e professores sobre a evasão nos Torneios ou como controlar. O tema evasão também não foi citado por nenhuma professor durante as entrevistas despadronizadas. Trabalhos futuros poderão estudar o que faz a evasão dos Torneios ser baixa em contraponto com a alta evasão em cursos/disciplinas da área de computação.

Seguindo o contexto do PBL e GBL presente no trabalho, focar nas novas tecnologias do GBL para realização de comparativos de melhoria de engajamento e desempenho escolar dos alunos participantes de competições com SG Minecraft. Adotar contextos de forma criativa a partir de tecnologias emergentes presentes no GBL, como as comunidades de aprendizagem, a partir de projetos que envolvam competições científicas ou oficinas que extrapolem o limite da escola. Avaliar tecnicamente os conceitos e práticas desenvolvidos e, principalmente, estudar a influência dos jogos digitais nessas comunidades de aprendizagem.

8. REFERÊNCIAS BIBLIOGRÁFICAS

Adams, D. M., Mayer, R. E., MacNamara, A., Koenig, A., & Wainess, R. (2012). Narrative Games for Learning: Testing the Discovery and Narrative Hypotheses. *Journal of Educational Psychology*, *104*(1), 235–249. http://doi.org/10.1037/a0025595

Alaiba, V., & Rotaru, A. (2008). Agent architecture for building robocode players with SWI-Prolog. *Proceedings of the International Multiconference on Computer Science and Information Technology, IMCSIT 2008*, *3*(2), 3–7. http://doi.org/10.1109/IMCSIT.2008.4747210

Assis, M. P. (2011). *Learning Design – Conceitos, Métodos e Ferramentas (Tese Doutorado)*. Doutorado em Educação. Pontifícia Universidade Católica de São Paulo – PUC-SP.

Aureliano, V. C. O., & Tedesco, P. C. D. A. R. (2012). Avaliando o uso do Scratch como abordagem Alternativa para o processo de ensino-aprendizagem de programação. *XX Workshop Sobre Educação Em Computação. XXXII CSBC*, 10. Retrieved from http://www.imago.ufpr.br/csbc2012/anais_csbc/eventos/wei/artigos/Avaliando o uso do Scratch como abordagem alternativa para o processo de ensino-aprendizagem de programacao.pdf

Ausubel, D. P. (1982). *Aprendizagem Significativa: A Teoria de David Ausubel*. São Paulo, SP: Moraes.

Ausubel, D. P. (2000). *The Acquisition and Retention of Knowledge: A Cognitive View*. New York: Springer Science and Business Media Dordrecht. http://doi.org/10.1007/978-94-015-9454-7

Ausubel, D. P., Novak, J. D., & Hanesian, H. (1978). *Educational Psychology: A Cognitive View* (2nd ed.). New York: Holt Rinehart and Winston.

Balzan, N. C. (1999). Formação de Professores para o Ensino Superior: Desafios e Experiências. *Bicudo MA, Organizador. Formação Do Educador E Avaliação Educacional. São Paulo: Editora UNESP*, 173–188.

Barnes, T., Powell, E., Chaffin, A., Goldwin, A., & Richter, H. (2007). Game2Learn : Building CS1 Learning Games for Retention. *ACM SIGCSE Bulletin*, *39*(3), 121–125. http://doi.org/10.1145/1268784.1268821

Barrows, H. S. (1996). Problem-based learning in Medicine and Beyond: a Brief Overview. In: Wilkerson, L.; Gijselaer, W. H. (Eds.). Bringing problem-based learning to higher education: theory and practice. San Francisco: Jossey-Bass, 3–12.

Becker, F. (1993). *A Epistemologia do Professor: o Cotidiano da Escola* (12th ed.). Petrópolis: Vozes.

Berbel, N. A. N. (1998). A problematização ea aprendizagem baseada em problemas. *Interface Comun Saúde Educ*, 139–154. http://doi.org/10.1590/S1414-32831998000100008

Blackman, S. (2005). Serious Games … and Less ! *Computer Graphics*, *39*(February), 12–16. http://doi.org/10.1145/1057792.1057802

Blikstein, P. (2006). *Avaliação da Aprendizagem em Educação On-line*. São Paulo, SP: Loyola.

Bonakdarian, E., & White, L. (2004). Robocode Throughout the Curriculum. *Journal of Computing Sciences in Colleges - JCSC. Consortium for Computing Sciences in Colleges - CCSC: Southeastern Conference.*, *19*(3), 311–313.

Borges, M. A. F. (2004). *Um Processo para Análise da Interação em Sistemas Colaborativos Mediados por Ferramentas Computacionais para Comunicação Textual (Tese Doutorado)*. Instituto de Computação – IC, Universidade Estadual de Campinas – UNICAMP, SP.

Briscoe, G., & Mulligan, C. (2014). Digital Innovation: The Hackathon Phenomenon. *Creativeworks London*, (6), 1–13.

Burd, L. (1999). *Desenvolvimento de Software para Atividades Educacionais (Dissertação de Mestrado)*. Departamento de Engenharia de Computação e Automação Industrial. Faculdade de Engenharia Elétrica e de Computação. Universidade Estadual de Campinas (UNICAMP). Retrieved from http://web.media.mit.edu/~leob/tese_total.pdf

Celso, R., Correia, M., Shimabukuro, M. H., & Simonsen, R. R. (2008). Ensino de Lógica de Programação e Estruturas de Dados para Alunos do Ensino Médio, 246–249.

Chaffin, A., Doran, K., Hicks, D., & Barnes, T. (2009). Experimental evaluation of teaching recursion in a video game. *Proceedings of the 2009 ACM SIGGRAPH Symposium on Video Games - Sandbox '09*, *1*(212), 79. http://doi.org/10.1145/1581073.1581086

Chaves, E. O. de C. (1998). *Tecnologia e Educação: O Futuro da Escola na Sociedade da Informação* (1st ed.). Campinas: Mindware.

Cheverton, M. (2014). *Invasion of the World. An Unofficial Minecrafter's Adventure*. New York: Sky Pony Press.

Clementino, A. (2008). *Didática Intercomunicativa em Cursos On-line Colaborativos. (Tese de Doutorado)*. Faculdade de Educação da Universidade de São Paulo - USP.

CNPq. (2016). Olimpíadas científicas. Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq). Retrieved July 16, 2016, from http://cnpq.br/olimpiadas-cientificas

Code.org. (2016). Non-profit Dedicated to Expanding Access to Computer Science. Retrieved July 1, 2016, from https://code.org/

Coelho, A. R. A. (2013). *Migração do Framework Furbot para a Plataforma Android (Monografia de Graduação)*. Universidade Regional de Blumenau – Centro de Ciências Exatas e Naturais – Bacharelado Ciência da Computação.

Cotta, A. (2002). *Novas Tecnologias Educacionais no Ensino de Matemática: Estudo de Caso – Logo e do Cabri-Geometre. (Dissertação de Mestrado)*. UFSC – Universidade Federal de Santa Catarina, Programa de Pós-graduação em Engenharia de Produção.

Dale, E. (1969). *Audio-Visual Methods in Teaching* (3rd ed.). New York: Holt, Rinehart e Winston.

Delisle, R. (2000). *Como Realizar a Aprendizagem Baseada em Problemas*. Lisboa, Portugal: Cadernos CRIAP - ASA. Edições Asa.

Digiampietri, L. A., Peres, S. M., Nakano, F., Roman, N. T., Wagner, P. K., Silva, B. B. C., … Barros, V. A. (2014). Complementando o Aprendizado em Programação: Revisitando Experiências no Curso de Sistemas de Informação da USP. *iSys-Revista Brasileira de Sistemas de Informação*, *6*(Sbsi), 779–790. Retrieved from http://www.seer.unirio.br/index.php/isys/article/view/2178

Duch, B. (2001). *Writing Problems for Deeper Understanding*. Sterling, Virginia: Stylus Publishing.

Eagle, M., & Barnes, T. (2008). Wu ' s Castle : Teaching Arrays and Loops in a Game. *Proceedings of the 13th Annual Conference on Innovation and Technology in Computer Science Education. Madri, Espanha*, *6*(1), 23–30.

Embrapa. (2016). Hackathon Embrapa Universitário. Manejo Integrado de Pragas para Sustentabilidade dos Agroecossistemas. Retrieved October 15, 2016, from https://www.embrapa.br/hackathon

Epistec. (2015). Intelligent Games. Introduction Ceebot, Have Fun Programming. Retrieved July 1, 2015, from http://www.ceebot.com

EPL. (2016). Eclipse Public License - v 1.0. Retrieved October 16, 2016, from http://www.eclipse.org/legal/epl-v10.html

Evans, T., & Nation, D. (2000). *Changing University Teaching: Reflections on Creating Educational Technologies (Open and Flexible Learning Series)*. London. UK: Kogan Page Limited.

Fassbinder, A. G. de O., Paula, L. C., & Araujo, J. C. D. (2012). Experiências no estímulo à prática de Programação através do desenvolvimento de atividades extracurriculares relacionadas com as competições de conhecimentos. *XX Workshop Sobre Educação Em Computação (WEI). XXXII Congresso Da Sociedade Brasileira Da Computação (CSBC)*, *10*.

Fayek, M. B., & Farag, O. S. (2015). HICMA: A human imitating cognitive modeling agent using statistical methods and evolutionary computation. *IEEE SSCI 2014 - 2014 IEEE Symposium Series on Computational Intelligence - CIHLI 2014: 2014 IEEE Symposium on Computational Intelligence for Human-Like Intelligence, Proceedings*. http://doi.org/10.1109/CIHLI.2014.7013383

FEPESP. (2011). Evasão na Educação Profissional. *Fórum Da Educação Profissional Do Estado de São Paulo - FEPESP, 31 de Maio de 2011, São Paulo. Anais Eletrônicos.*, 1–1. Retrieved from http://www.cpscetec.com.br/fepesp/anteriores.html

Ferreira, B. J. P., & Duarte, N. (2012). Lema de Aprender a Aprender na Literatura de Informática Educativa. *Educação & Sociedade. Centro de Estudos Educação E Sociedade Brasil, Campinas, SP*, *33*(121), 1019–1035. Retrieved from http://www.redalyc.org/articulo.oa?id=87325199006

Filho, E. E., & Ribeiro, L. R. de C. (2009). Aprendendo com PBL – Aprendizagem Baseada em Problemas: relato de uma experiência em cursos de engenharia da EESC-USP. *Revista Minerva–Pesquisa E Tecnologia*, *6*(1), 23–30.

Fisher, E. (1993). *The teacher's role. In: Language, Classrooms and Computers. Scrimshaw, P. (Ed.)*. London. UK: Routledge.

Furbot. (2015). Project: A Framework to Facilitate the Learning of Computer Programming with Java. Retrieved September 1, 2015, from http://sourceforge.net/projects/furbot

Gallant, R. J., & Mahmoud, Q. H. (2008). Using Greenfoot and a moon scenario to teach Java programming in CS1. *Proceedings of the 46th Annual Southeast Regional Conference on XX, ACM-SE 46*, 118–121. http://doi.org/10.1145/1593105.1593135

Garris, R., & Driskell, J. E. (2002). Games , Motivation , and Learning : A Research and Practice Model Robert Ahlers. *Simulation and Gaming*, *33*, 441–467. http://doi.org/10.1177/1046878102238607

Gee, J. P. (2004). Learning by design: Games as learning machines. *Interactive Educational Multimedia*, *8*(8), 15–23. http://doi.org/10.2304/elea.2005.2.1.5

Gee, J. P. (2007). *What Video Games Have to Teach Us About Learning and Literacy* (2nd ed.). New York: St. Martin's Griffin.

Gee, J. P. (2008). Game-like Learning: An Example of Situated Learning an Implications for Opportunity to Lean. *Assessment, Equity, and Opportunity to Learn. Cambridge, UK. Cambridge University Press*, 200–221.

Gee, J. P. (2013). *Good Video Games and Good Learning: Collected Essays on Video Games, Learning and Literacy.* (2nd ed.). New York: Peter Lang Publishing Inc.

Gil, A. C. (2008). *Métodos e Técnicas de Pesquisa Social* (6th ed.). São Paulo: Atlas.

Gros, B. (2003). The impact of digital games in education. *Frist Monday*, *8*(7), 1689–1699. http://doi.org/10.1017/CBO9781107415324.004

Guinness. (2014). Officially Guinness World Records. Retrieved September 1, 2015, from http://www.guinnessworldrecords.com

Hakulinen, L. (2011). Survey on Informatics Competitions : Developing Tasks. *Olympiads in Informatics*, *5*, 12–25.

Haycock, K. (2007). Collaboration : Critical success factors for student learning. *School Libraries Worldwide . School of Library and Information Science, San José State University, USA*, *13*(1), 25–35. Retrieved from http://scholarworks.sjsu.edu/slis_pub

Hmelo-Silver, C. E., & Barrows, H. S. (2006). Goals and Strategies of a Problem-based Learning Facilitator. *The Interdisciplinary Journal of Problem-Based Learning*, *1*(1), 21–39. http://doi.org/10.7771/1541-5015.1004

Hong, J., & Cho, S.-B. (2004). Evolution of emergent behaviors for shooting game characters in Robocode. *Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No.04TH8753)*. http://doi.org/10.1109/CEC.2004.1330917

Hou, S.-I. (2014). Integrating Problem-Based Learning with Community-Engaged Learning in Teaching Program Development and Implementation. *Universal Journal of Educational Research*, *2*(1), 1–9. http://doi.org/10.13189/ujer.2014.020101

Howland, G. (1999). The Focus of Gameplay. Retrieved July 1, 2015, from http://archive.gamedev.net

ICMC-USP. (2012). Competições científicas. Instituto de Ciências Matemáticas e de Computação (ICMC) da Universidade de São Paulo (USP). *ICMCotidiano - Ano XIII, nº99, Outubro/Novembro/Dezembro - 2012*, 1–20. Retrieved from http://www.icmc.usp.br/CMS/Arquivos/arquivos_enviados/ADMINISTRADOR_113_cotidiano99.pdf

Jenkins, T. (1998). A participative approach to teaching programming. *ACM SIGCSE Bulletin*, *30*(3), 125–129. http://doi.org/10.1145/290320.283090

Jiménez, K. C. (2009). Evaluación cualitativa y gestión del conocimiento. Educación y Educadores. *Educación Y Educadores. Chia. Qualitative Assessment and Management*, *12*(3), 179–195. Retrieved from http://www.scielo.org.co/pdf/eded/v12n3/v12n3a10

Johnson, W. L. (2010). Serious use of a serious game for language learning. *International Journal of Artificial Intelligence in Education*, *20*(2), 175–195. http://doi.org/10.3233/JAI-2010-0006

Kalelioglu, F., & Gulbahar, Y. (2014). The effects of teaching programming via Scratch on problem solving skills: A discussion from learners' perspective. *Informatics in Education*, *13*(1), 33–50.

Kasvi, J. J. J. (2000). Not Just Fun and Games - Internet Games as a Training Medium. *Cosiga - Learning With Computerised Simulation Games. HUT: Espoo. Finland: Helsinki University of Technology Laboratory of Work Psychology*, 23–34.

Kitano, H., & Veloso, M. (1997). No The RoboCup Synthetic Agent Challenge 97. *XV IJCAI-97 International Joint Conference on Artificial Intelligence*, *1*, 24–29.

Kobayashi, K., Uchida, Y., & Watanabe, K. (2003). A study of battle strategy for the Robocode. *SICE Annual Conference in FUkui*, *7*(2), 4–6. Retrieved from http://www.wtnb.k.hosei.ac.jp/html/article/ronbun/03ronbun/021-Uchida.pdf

Kolodner, J. L., Hmelo-Silver, C. E., & Narayanan, N. H. (1996). Problem-based learning meets case-based reasoning, 188–195. Retrieved from http://dl.acm.org/citation.cfm?id=1161161

Koper, R., & Tattersall, C. (2005). *Learning Design: A Handbook on Modelling and Delivering Networked Education and Training*. Heidelberg: Springer-Verlag.

Kwon, S. Y., & Cifuentes, L. (2009). The comparative effect of individually-constructed vs. collaboratively-constructed computer-based concept maps. *Computers and Education*, *52*(2), 365–375. http://doi.org/10.1016/j.compedu.2008.09.012

Li, S. (2002). Rock 'em, sock 'em Robocode!, Learning Java programming is more fun than ever with this advanced robot battle simulation engine. Retrieved January 1, 2015, from http://www.ibm.com/developerworks/java/library/j-robocode

Lima, M. S. S. (2016). *Ferramentas e Métodos para Organizar e Perpetuar uma Competição Nacional de Robocode (Monografia de Graduação)*. Bacharelado de sistemas de informação - Faculdade de Tecnologia - FT - Universidade Estadual de Campinas - UNICAMP.

Maloney, J., Resnick, M., & Rusk, N. (2010). The Scratch programming language and environment. *ACM Transactions on Computing Education*, *10*(4), 1–15. http://doi.org/10.1145/1868358.1868363.http

Maluf, A. C. M. (2008). *Brincar - Prazer e Aprendizado* (6th ed.). Petrópolis: Vozes.

Marconi, M. A., & Lakatos, E. M. (2010). *Fundamentos de Metodologia Científica* (7th ed.). São Paulo: Atlas.

Marques, F. O., & Marques, M. T. (2012). No problem ? No research, little learning ...Big problem! *Journal of Systematics, Cybernetics & Informatics*, *10*(3), 60–63.

Matsunaga, R. M., Moraes, R. L. de O., Borges, M. A. F., Matta, M. A. P., & Ozelo, M. C. (2014). Development of a Serious Game for children with hemophilia Desenvolvimento de um Serious Game para crianças com hemofilia Desarrollo de un Serious Game para niños con hemofilia. *J. Health Inform*, *6*, 114–9.

Mattar, J. (2010). *Games em Educação: Como os Nativos Digitais Aprendem* (1st ed.). São Paulo, SP: Pearson, Prentice Hall.

Matumoto, C. A., Poli, G. A., Oliveira, E. S., & Borges, G. A. (2015). *Proposta de Curso para Emsino de Linguagem de Programação com Jogo Educacional Robocode (Trabalho de Conclusão de Curso)*. Técnico em Informática Integrado ao Ensino Médio - Instituto Federal de São Paulo - IFSP.

Meira, M. C., Lima, M. S. S., & Borges, M. A. F. (2016). Torneios Baseados em Robocode para Incentivar Jovens a Aprender Programação. *Anais Dos Workshops Do XXXVI Congresso Da Sociedade Brasileira de Computação (CSBC 2016)*, (CSBC), 2403–2412. Retrieved from http://www.pucrs.br/edipucrs/

MercadoLibre, D. (2016). Hackathon Mercado Libre Developers. Retrieved October 1, 2016, from http://developers.mercadolibre.com/events/hackathon-mercado-libre-microsoft/

Mezzari, A. (2011). O uso da Aprendizagem Baseada em Problemas (ABP) como reforço ao ensino presencial utilizando o ambiente de aprendizagem Moodle. *Revista Brasileira de Educação Médica*, *35*(1), 114–121. http://doi.org/10.1590/S0100-55022011000100016

Nidorf, D. G., Barone, L., & French, T. (2010). A comparative study of NEAT and XCS in robocode. *2010 IEEE World Congress on Computational Intelligence, WCCI 2010 - 2010 IEEE Congress on Evolutionary Computation, CEC 2010*, 1–8. http://doi.org/10.1109/CEC.2010.5586087

Nishimura, T., Kawasaki, S., & Tominaga, H. (2011). Monitoring system of student situation in introductory C programming exercise with a contest style. *2011 International Conference on Information Technology Based Higher Education and Training, ITHET 2011*. http://doi.org/10.1109/ITHET.2011.6018693

NMC, P. H. (2012). Perspectivas Tecnológicas para o Ensino Fundamental e Médio Brasileiro de 2012 a 2017. Retrieved August 1, 2015, from http://zerohora.com.br/pdf/14441735.pdf

Novak, J. D., & Gowin, B. (1996). *Aprender a Aprender* (2nd ed.). Lisboa: Plátano Edições Técnicas.

Nunes, J. M., & Infante, M. (1996). *Pesquisa-ação: uma metodologia de consutoria*. Rio de Janeiro: Fiocruz - SciELO Books.

O'Kelly, J., & Gibson, J. (2006). RoboCode & problem-based learning: a non-prescriptive approach to teaching programming. *ACM SIGCSE Bulletin*, *38*(June), 217–221.

http://doi.org/10.1145/1140123.1140182

Oliveira, M. L. S. de, Souza, A. A. de, Barbosa, A. F., & Barreiros, E. F. S. (2014). Ensino de lógica de programação no ensino fundamental utilizando o Scratch: um relato de experiência. *Congresso Da Sociedade Brasileira de Computação – CSBC*, *12*(2006), 1–10. Retrieved from http://www.lbd.dcc.ufmg.br/colecoes/wei/2014/0022.pdf

Papert, S. (1986). Constructionism: A New Opportunity for Elementary Science Education. *A Proposal to the National Science Foundation. Cambridge Massachusetts Institute of Technology (MIT), Media Laboratory, Epistemology and Learning Group.*

Papert, S. (1993). *The Children's Machine: Rethinking School In The Age Of The Computer*. New York: Basic Books; Revised ed. edition.

Papert, S. (1996). *The Connected Family: Bridging the Digital Generation Gap*. Atlanta: Taylor Trade Publishing.

Pereira, T. G., Costa, E. A. J., Sales, M. B. S., & Sales, A. B. (2016). Utilização de Problemas da Maratona de Programação e Juízes Eletrônicos como Estratégia de Ensino em um Curso de Graduação em Engenharia de Software. *V Congresso Brasileiro de Informática Na Educação (CBIE 2016). Anais Do XXVII Simpósio Brasileiro de Informática Na Educação (SBIE 2016)*, (Cbie), 210–219. http://doi.org/10.5753/cbie.sbie.2016.210

Piaget, J. A. (1978). *A Formação do Símbolo na Criança* (3rd ed.). Rio de Janeiro - RJ: Zahar.

Piekarski, A. E., Miazaki, M., Hild, T., Mulati, M. H., & Kikuti, D. (2015). A metodologia das maratonas de programação em um projeto de extensão: um relato de experiência, (Cbie), 1246. http://doi.org/10.5753/cbie.wcbie.2015.1246

Prado, C. (2009). *Cultura Digital.br. Organização Rodrigo Savazoni, Sergio Cohn*. Rio de Janeiro - RJ: Azougue Editorial.

Prensky, M. (2001). Digital Natives, Digital Immigrants. *From On the Horizon MCB University Press*, *9*(5), 1–6.

Prensky, M. (2012). *Aprendizagem Baseada em Jogos Digitais*. (E. Yamagute & R. Tori, Eds.). São Paulo: Senac São Paulo.

Recchia, T., Chung, J., & Pochiraju, K. (2014). Performance of heterogeneous robot teams with personality adjusted learning. *Biologically Inspired Cognitive Architectures*, *7*, 87–97. http://doi.org/10.1016/j.bica.2013.10.003

Rhem, J. (1998). Problem-Based Learning : An Introduction. *The Nacional Teching & Learning Forum*, *8*(1), 1–7.

Richard, G. T., Kafai, Y. B., Adleberg, B., & Telhan, O. (2015). StitchFest: diversifying a

college hackathon to broaden participation and perceptions in computing. In *SIGCSE '15: Proceedings of the 46th ACM Technical Symposium on Computer Science Education* (pp. 114–119). http://doi.org/10.1145/2676723.2677310

Robbins, S. P. (2011). *Comportamento Organizacional* (14th ed.). São Paulo, SP: Prentice Hall.

Robocode. (2013). Open Source Educational Game, ReadMe for Robocode. Retrieved June 1, 2015, from http://robocode.sourceforge.net

Robocup. (2016). The RoboCup Federation. Retrieved October 1, 2016, from http://www.robocup2016.org

RoboWiki. (2015). Collecting Robocode Knowledge Since 2003. Retrieved July 1, 2015, from http://http//robowiki.net

Rodriguez, J. A. H. (2015). *Start-up Development in Latin America: The Role of Venture Accelerators (Masters dissertation)*. Massachusetts Institute of Technology - MIT, Sloan School of Management.

Santoro, F., Borges, M. R. da S., & Santos, N. (1999). Um framework para estudo de ambientes de suporte à aprendizagem cooperativa. *Revista Brasileira de Informática Na Educação*, *4*(2), 51–68. Retrieved from http://ceie-sbc.educacao.ws/pub/index.php/rbie/article/view/2293

Saviani, D. (2012). *Escola e Democracia* (42nd ed.). Campinas: Autores Associados.

SBC. (2016). Maratona de Programação da Sociedade Brasileira da Computação (SBC). Retrieved September 30, 2016, from http://www.sbc.org.br/educacao/maratona-de-programacao

Schuytema, P. (2008). *Design de Games: Uma Abordagem Prática*. São Paulo, SP: Cengage Learning.

ScienceOlympiad. (2016). Science Olympiad. Retrieved October 10, 2016, from https://www.soinc.org

Scratch. (2015). Create Stories, Games, and Animations Share With Others Around the World. Retrieved October 1, 2015, from http://scratch.mit.edu

Shaffer, D. W. (2006). *How Computer Games Help Children Learn*. New York: Palgrave.

Skinner, B. F. (1993). *Ciência e Comportamento Humano*. São Paulo, SP: Martins Fontes.

Soeira, E. dos R., & Schneider, H. N. (2013). Ead: Percepções de Tutores a Distância. *International Journal of Knowledge Engineering and Management. ISSN 2316-6517, Florianópolis, Santa Catarina*, *2*(4), 109–134.

Stahl, M. M. (1990). Software Educacional: Características dos Tipos Básicos. *In: I Simpósio Brasileiro de Informática Na Educação. Anais. Rio de Janeiro.*, 34–45.

Stallman, R. (2014). Por Que Escolas Devem Usar Exclusivamente Software Livre. Retrieved July 1, 2015, from http://www.gnu.org/education/edu-schools.pt-br.html

Stone, P. (2016). What ' s Hot at RoboCup (Extended Abstract). *Proceedings of the 30th Conference on Artificial Intelligence (AAAI 2016)*, 4346–4347.

Susi, T., Johannesson, M., & Backlund, P. (2007). Serious Games – An Overview. *Elearning*, *73*(10), 28. http://doi.org/10.1.1.105.7828

Tang, S., Hanneghan, M., & El Rhalibi, A. (2009). *A. Introduction to Games Based Learning. In T. Connolly, M. Stansfield and L. Boyle (eds.), Games Based Learning Advancements for MultiSensory Human Computer Interfaces*. New York: IGI Global.

Thiollent, M. (2008). *Metodologia de Pesquisa-Ação* (18th ed.). São Paulo: Cortez.

Tobias, S., Fletcher, J. ., Dai, D. ., & Wind, A. P. (2011). Review of Research on Computer Games. In *Computer games and instruction.* (pp. 127–222). Charlotte: Information Age.

Torres, P. L., & Irala, E. A. F. (2007). *Aprendizagem Colaborativa. In: TORRES, P. L. (org.). Algumas Vias para Entretecer o Pensar e o Agir*. Curitiba, PR: SENAR-PR, 65-98.

Trescastro, L. B., & Maria, C. (2009). Blog : Aprendizagem Interativa na Formação Continuada de Professores. *15º CIAED, Congresso Internacional ABED, Associação Brasileira de Educação a Distância, ISBN: 2175-4098, Fortaleza*, 1–10.

Valente, J. A. (1999). *O Computador na Sociedade do Conhecimento*. Campinas, São Paulo: OEA_NIED/UNICAMP.

Vanbuel, M. (1998). Blueprint for the Interactive Classroom. *Educational Media International*, *35*(1), 18–20. http://doi.org/10.1080/0952398980350106

Vannucchi, H., & Prado, G. (2009). Discutindo o conceito de Gameplay. *Texto Digital*, *5*(2), 130–140. http://doi.org/10.5007/1807-9288.2009v5n2p130

Winslow, L. E. (1996). Programming Pedagogy - A Psychological Overview. *ACM SIGCSE Bulletin*, *28*(3), 17–22. http://doi.org/10.1145/234867.234872

Woot!Craft. (2015). Woot! Academy. Curso de programação Woot!Craft com Minecraft. Retrieved July 1, 2015, from http://www.wootacademy.com.br

Yessad, A., Labat, J. M., & Kermorvant, F. (2010). SeGAE: A serious game authoring environment. *Proceedings - 10th IEEE International Conference on Advanced Learning Technologies, ICALT 2010*, 538–540. http://doi.org/10.1109/ICALT.2010.153

Zyda, M. (2005). From visual simulation to virtual reality to games. *Computer*, *38*(9), 25–32. http://doi.org/10.1109/MC.2005.297

APÊNDICES

I. Apêndice – Ações de Programação Robocode

Ações de programação do robô são conduzidas por um conjunto de comandos da API do Robocode, documentados com Javadoc. A maioria está relacionada com o movimento do robô, canhão e radar. A Tabela 1 exibe ações básicas definidas para programação do tanque.

Tabela 1- Métodos Básicos para Movimento (Hong & Cho, 2004).

Comando	Parâmetro	Descrição
turnRight(double) *turnLeft(double)*	Ângulo que o robô deverá girar.	Gira o robô para a direita ou esquerda em graus.
ahead(double) *back(double)*	Distância que o robô deverá percorrer.	Movimenta o robô para frente ou traz, uma distância (x) dada por parâmetro em pixel. Se o robô bater em outro, ou na parede antes de completar a distância desejada o método é interrompido.
turnGunLeft(double) *turnRadarRigth(double)*	Ângulo que o canhão deverá girar	Gira o canhão para a direita ou esquerda em graus.
turnRadarRigth(double) *turnRadarLeft(double)*	Ângulo em graus que o radar deverá girar	Gira o radar para a direita ou esquerda em graus.

A Tabela 2 indica a possibilidade de deslocar, simultaneamente, o canhão e o radar com a movimentação do veículo através da chamada de métodos específicos para cada ação.

Tabela 2 - Métodos para Canhão e Radar (Hong & Cho, 2004).

Comando	Descrição
setAdjustGunForRobotTurn(boolean)	Se definido como verdadeiro, a arma vai permanecer na mesma direção quando o veículo se vira.
setAdjustRadarForRobotTurn(boolean)	Se definido verdadeiro, o radar permanecerá na direção enquanto o veículo (e a canhão) gira.
setAdjustRadarForGunTurn(boolean)	Se definido como verdadeiro, o radar permanecerá na mesma direção enquanto o canhão gira.

Com objetivo de definir a estratégia do comportamento do robô, é possível coletar informações durante a batalha. Com esse propósito, Robocode fornece alguns métodos para obter informações do robô. A Tabela 3 mostra uma lista com os métodos mais usuais.

Com movimentação e canhão associados, deve ser estabelecida uma estratégia para disparos de forma adequada (Hong & Cho, 2004). Cada robô começa com uma energia padrão

do canhão. Quando atinge o valor 0 (zero), ele cessa o disparo (método *fire()*), tornando-se um alvo fácil para adversários. É possível utilizar, em um disparo, até 3 (três) unidades de energia com passagem de parâmetro ao método *fire()*. Quanto mais energia fornecida ao disparo, maior o dano causado ao adversário. *Fire (double power)* e *fireBullet (double power)* são métodos básicos para disparar com especificação de energia.

Tabela 3- Métodos básicos para obtenção de informações (Hong & Cho, 2004).

Comando	Retorno	Descrição
getX() *getY()*	*double*	Retorna a posição X (eixo horizontal) ou Y (eixo vertical) do robô na arena de batalha. Quando X=0(zero) ele estará encostado no lado esquerdo. Quando Y=0(zero) ele estará encostado na parte inferior.
getHeading() *getGunHeading()* *getRadarHeading()*	*double*	Retorna o ângulo em graus (0 até 360) que o robô, canhão e radar estão virados respectivamente. Se retornar 0(zero) ele está virado para a esquerda, se retornar 90 ele está voltado para cima.
getBattleFieldHeight() *getBattleFieldWidth()*	*double*	Retorna à altura e largura da arena de batalha.

A movimentação do robô ocorre de acordo com os eventos. A classe básica do robô (*Robot class*) possui manipuladores padrões para os eventos. Os eventos são chamados quando ocorrem ações específicas no decorrer do combate. Alguns retornam dados do robô, como ângulo, ao colidir com a parede. Na ocorrência de eventos como *onHitWall()*, devem haver ações para mudar a direção do robô, para evitar a perda de energia com a colisão. A Tabela 4 exibe alguns destes eventos frequentemente utilizados.

Tabela 4 - Eventos Básicos do robô no Robocode (Hong & Cho, 2004).

Comando	Executado quando:
onScannedRobot ()	O radar do robô encontra um adversário.
onHitByBullet ()	O robô leva um tiro.
onHitRobot ()	Robô bate em outro robô.
onHitWall()	Robô colide com a parede.
onBulletHit()	Tiro acerta um adversário.

Com o início da disputa, o robô se movimenta com base em um comportamento básico da classe principal (*main loop*) e os comportamentos adicionais ocorrem conforme a ocorrência dos eventos. Eventos como *HitByBulletEvent*, *HitRobotEvent*, *HitBulletEvent*, *HitWallEvent*, etc. podem ser utilizados para programação dos robôs. Existem quatro tipos de robôs no Robocode, apresentados na Tabela 5.

Tabela 5 - Tipos de Robôs no Robocode.

Tipo de Robô	Características
JuniorRobot	Propósitos educacionais, utiliza o *package junior.* Possui principais funcionalidades prontas (radar, tiro, movimentação etc.). Os comentários padrões do *JuniorRobot* possuem mais detalhes em relação ao *Robot*.
Robot	Com principais funcionalidades prontas (radar, disparo, movimentação etc.) e comentários. Utiliza o *package robot.*
AdvancedRobot	Permite desenvolvimento complexo para alteração ou criação de funcionalidades.
Droid	Robôs não possuem radar, disparam a partir da orientação de outros robôs. Recebem bônus de energia por não possuírem radar.

A partir da programação de um *Robot*, o mesmo se movimenta com base no fluxo exibido na Figura 1. As estratégias para os eventos são decididas com a seleção de um determinado comportamento, conforme a variação de dados coletados durante a partida. Contudo, a complexidade e combinações dos comportamentos tende a aumentar rapidamente os fluxos da estrutura, conforme a criação e inserção de novos eventos.

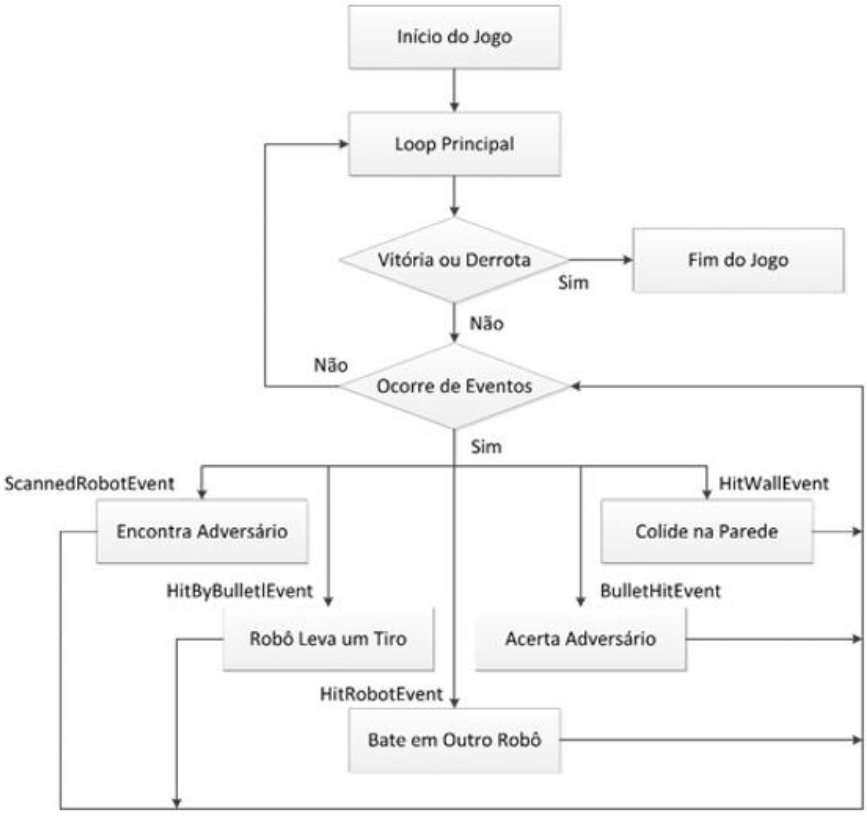


Figura 1 - Fluxo de Comportamento Básico de um Robô (Hong & Cho, 2004).

II. Apêndice – Detalhamento das Regras do Robocode

Os robôs perdem energia a partir das seguintes situações: (1) Ao receber uma bala, a energia é perdida conforme a potência do disparo adversário; (2) Danos descontados aos envolvidos no valor de 1 (um) quando existe impacto entre robôs; (3) Quando bater na parede a energia perdida tem sua variação conforme a velocidade do impacto. Quando acertar um adversário (*hit*), o robô que disparou recebe um pouco da energia retirada do robô que levou a bala. As balas possuem um movimento linear, com velocidade inversamente proporcional à potência do disparo (Robocode, 2013).

Existe também o conceito de "calor", que é gerado no canhão, em um disparo. Quando o canhão dispara, é gerado calor igual a 1 + *fire(x)*/5 e, se o valor do calor ultrapassar 3 (três), o canhão poderá ficar inutilizado (não poderá disparar novamente) até que esfrie. Esse fato ocorre quando vários disparos consecutivos são realizados, geralmente em potência máxima de disparo (*fire* = 3), em um curto intervalo de tempo (Robocode, 2013).

O comando denominado *fire(x)* é responsável por disparar nos adversários, onde *x* é a potência da bala (assume valores de 1 a 3). A Tabela 1 e a Tabela 2 indicam algumas regras aplicadas ao jogo, para quando se acerta o adversário (*hit*) ou quando se desperdiça uma bala (*miss*) respectivamente. É fundamental que o programador conheça as regras do jogo para a definição da estratégia dos robôs conforme as situações apresentadas durante a partida.

Tabela 1 - Pontuação Robocode, *hit* (Robocode, 2013).

Potência (P)	**Danos ao Adversário**	**Ganho Atacante**
P1	Perde 4 pontos de energia	Ganha 2 pontos de energia
P2	Perde 8 pontos de energia	Ganha 4 pontos de energia
P3	Perde 16 pontos de energia	Ganha 6 pontos de energia

Tabela 2 - Regras de Pontuação Robocode, *miss* (Robocode, 2013).

Potência (P)	**Danos ao Atacante**
P1	Perde 1 pontos de energia
P2	Perde 2 pontos de energia
P3	Perde 3 pontos de energia

É possível citar três componentes primários de estratégias inseridas para os desenvolvedores de robôs no Robocode: (1) Ao encontrar um adversário, a partir da identificação com radar, ajustar o alvo e efetuar disparos; (2) Mover-se por todo campo de batalha, com objetivo de desviar das balas; (3) Tratar o posicionamento do robô a fim de coletar dados para melhor orientação. Cada um destes componentes caracteriza, respectivamente, os módulos de: *scanning*; *targeting* e *movement*, que podem ser programados e testados separadamente. Os robôs devem ser competentes em todas as áreas com objetivo atingir máxima proficiência (Nidorf et al., 2010).

Existe a possibilidade da realização da análise comportamental do agente adversário durante as disputas. Estratégias podem ser definidas com base na predição da movimentação linear ou previsão da bala do agente adversário. Avaliações que sigam estas características podem garantir um agente competitivo no ambiente Robocode (Kobayashi, Uchida, & Watanabe, 2003).

III. Apêndice – Torneios no Brasil nos Últimos Anos

Instituição	Cidades	Ano(s) do Torneio(s)
COTIL-UNICAMP - Colégio Técnico de Limeira	Limeira-SP	2015, 2014, 2013, 2012, 2011, 2010
FT-UNICAMP - Faculdade de Tecnologia	Limeira-SP	2015, 2014, 2013, 2012, 2011, 2010
ETEC Centro Paula Souza - João Belarmino	Amparo-SP	2014, 2013, 2012, 2011
IFSP - Instituto Federal São Paulo	Capivari-SP	2015, 2014, 2011
IFAL - Instituto Federal de Alagoas	Arapiraca-AL	2015, 2014
IF Farroupilha	São Borja-RS	2015, 2014
UFU - Universidade Federal de Uberlândia	Patos de Minas-MG	2013, 2012
UFMA - Universidade Federal do Maranhão	São Luís-MA	2014, 2009
UFC - Universidade Federal do Ceará	Quixadá-CE	2011, 2007
ETEC Centro Paula Souza - Armando Pannunzio	Sorocaba-SP	2015
ETEC Centro Paula Souza - Fernando Prestes	Sorocaba-SP	2015
ETEC Centro Paula Souza Adolpho Berezin	Mongaguá-SP	2015
IESP Instituto de Ensino Superior da Paraíba	Cabedelo-PB	2015
IFSP - Instituto Federal São Paulo	Boituva-SP	2015
CETEC Santa Efigênia Centro Paula Souza São Paulo	São Paulo-SP	2014
ETEC Centro Paula Souza - Lauro Gomes	São Bernardo do Campo-SP	2014
ETEC Centro Paula Souza - Prof. José Ignácio Azevedo	Ituverava-SP	2014
ETEC Centro Paula Souza José Luiz Viana Coutinho	Jales-SP	2014
ETEC Centro Paula Souza Prof. Mário Antônio Verza	Palmital-SP	2014
IBTA - Instituto Brasileiro de Tecnologia Avançada -	São Paulo-SP	2014
IFG - Instituto Federal Goiás	Luziânia-GO	2014
IFSC - Instituto Federal Santa Catarina	São José-SC	2014
IFSP - Instituto Federal São Paulo	Campus do Jordão-SP	2014
UNIPAR - Universidade Paranaense	Francisco Beltrão-PR	2014
IFPI - Instituto Federal Piauí	Floriano-PI	2014
IFBA - Instituto Federal Bahia	Vitória da Conquista-BA	2013
IFRN - Instituto Federal do Rio Grande do Norte	Nova Cruz-RN	2013
UDESC - Universidade do Estado de Santa Catarina	Joinville-SC	2013
UFU - Universidade Federal de Uberlândia	Uberlândia-MG	2013
UNIFAL - Universidade Federal de Alfenas	Alfenas-MG	2013
UTFPR - Universidade Tecnológica Federal do Paraná	Pato Branco-PR	2013
Colégio São Luís	Dores do Indaiá-MG	2012
ETEC Centro Paula Souza Olímpia	Olímpia-SP	2012
Faculdade Joaquim Nabuco	Paulista-PE	2012
IC-UNICAMP - Instituto de Computação	Campinas-SP	2012
IFC - Instituto Federal Catarinense	Camburiú-SC	2012
IFC Instituto Federal Sul-Rio-Grandense	Bajé-RS	2012
IFES Instituto Federal Espírito Santo	Colatina-ES	2012
UNISAL - Centro Universitário Salesiano de São Paulo	Lorena-SP	2012
URI - Universidade Regional Integrada do Alto Uruguai	Santo Ângelo-RS	2012
FAAR - Faculdades Associadas de Ariquemes	Ariquemes-RO	2011
IFPR - Instituto Federal Paraná	Umuarama-PR	2011
UDESC - Universidade do Estado de Santa Catarina	Ibirama-SC	2011
UNIPAC - Faculdade Presidente Antônio Carlos	Contagem-MG	2011
Estácio - Centro Universitário UniSEB	Ribeirão Preto-SP	2010
FAGOC - Faculdade Governador Ozanam Coelho	Ubá-MG	2008
PUC Minas - Pontifícia Universidade Católica	Guanhães-MG	2008
UFC - Universidade Federal do Ceará - Campus Pici	Fortaleza-CE	2007

IV. Apêndice – Edital Robocode Brasil

EDITAL

I LIGA NACIONAL DE ROBOCODE

A Liga Robocode 2016 constitui a I Competição Nacional de Robocode cadastrada no repositório de desenvolvimento oficial do Robocode (http://robocode.sourceforge.net em *Upcoming Competitions* Liga Brasileira de Robocode at Unicamp University of Brazil). As competições são desenvolvidas e gerenciadas pelo LIAG (Laboratório de Informática, Aprendizado e Gestão, http://www.ft.unicamp.br/liag) da FT-UNICAMP (Faculdade de Tecnologia da Universidade Estadual de Campinas) desde 2010 com torneios e ligas interinstitucionais.

A competição terá início em 08/08/2016 e término em 17/10/2016 e será dividida em duas etapas: Torneio Local (08/08/2016 à 19/09/2016) e Liga Nacional (26/09/2016 à 17/10/2016). Os Torneios Locais serão sediados internamente na Instituição cadastrada. O campeão de cada Torneio Local disputará a Liga Nacional Robode 2016.

As instituições interessadas devem solicitar a inscrição (ITEM 1) através do e-mail contato@robocode.com.br ou selecionar no site (em inscrições http://www.robocodebrasil.com.br) a opção "Minha instituição ainda não está cadastrada" (ex. solicitar cadastro da "Instituição IF Farroupilha – Campus São Borja"). Após o cadastro da instituição, será possível cadastrar equipes, ITEM 2 (ex. a "Instituição IF Farroupilha – Campus São Borja" poderá cadastrar equipes).

Cada instituição participante da competição nacional Robocode 2016 deverá sediar um torneio local. Para sediar um torneio local a Instituição deverá inscrever um mínimo de 4 equipes. O Torneio Local deverá ser acompanhado por um representante da Instituição. Essas equipes irão se enfrentar de acordo com o ITEM 3.

A equipe campeã do Torneio Local disputa a Liga Nacional com equipes campeãs de outras Instituições (ex. semifinais de IF Farroupilha x FT-UNICAMP e IFSP x COTIL-UNICAMP).

Robocode é um jogo de programação com objetivo de codificar batalhas para competir em uma arena com outros robôs. O jogador é o programador do robô virtual, cuja função é desenvolver a lógica e estratégias com informações de comportamento e reação a eventos que ocorrem na arena. Os programadores têm a função de elaborar a inteligência artificial de cada robô a fim de estabelecer seu comportamento frente eventos que acontecerem na arena. As batalhas são executadas em tempo real e com representação gráfica. O software é composto por um ambiente completo de desenvolvimento: instalador próprio, editor de robôs e compilador Java/C#.

Resumo da Competição:

- Verificar as datas das Inscrições;
- Cadastrar Instituição no Torneio Local;
- Representante da Instituição cadastrar as equipes (mínimo de 4 equipes)*;
  *Caso alguma Instituição tenha interesse em participar e não possua 4 equipes interessadas poderá inscrever no Torneio "Público".
- Torneio Local terá início em 08/08/2016;
- Equipes devem enviar seus códigos para cada rodada.
- Representante local informa na plataforma http://www.robocodebrasil.com.br as pontuações das equipes em cada rodada;
- Liga Nacional terá início em 26/09/2016;
- Ao final do Torneio Local a equipe campeã disputará a Liga Nacional*;
  * Instituições com mais de 8 equipes cadastradas, terão DUAS vagas para disputar a Liga Nacional (1º e 2º colocados respectivamente terão acesso).
- Liga Nacional será administrada pela equipa do LIAG da FT-UNICAMP.

1. Inscrições

1.1. As inscrições ocorrerão entre os dias 04/04/2016 a 30/06/2016.

1.2. A Instituição (Escola, Faculdade, Bairro, Clube) com interesse em receber um Torneio Local deverá entrar em contato no e-mail: contato@robocode.com.br ou selecionar no site (em inscrições http://www.robocodebrasil.com.br) a opção "Minha instituição ainda não está cadastrada". No corpo do e-mail deverá constar o nome completo Instituição e nome do representante da Instituição. Indicar a função exercida dentro da instituição (ex. Professor de robótica, linguagem de programação etc., pedagogo, aluno do curso de Sistemas de Informação, etc.).

1.3. O responsável de cada Instituição tem a função de administrar o Torneio Local: divulgar o Torneio; divulgar as rodadas; organizar as partidas (pontos corridos, semifinal e final). O representante local receberá um usuário e senha para gerenciar o Torneio Local (estará disponível em http://www.robocodebrasil.com.br). A cada partida, o responsável deverá lançar os resultados no site oficial da competição.

1.4. Para um torneio ser válido é necessário que haja no mínimo 4 equipes vinculadas a este torneio.

Obs.: No encerramento das inscrições, equipes que estiverem cadastradas em Torneios Locais com menos de 4 equipes inscritas, serão realocadas para o Torneio "Público". Neste caso, o responsável pelo Torneio serão os administradores da Liga.

2. Equipes

2.1. Para se inscrever, a equipe deverá estar relacionada a um Torneio já cadastrado.

2.2. Equipes devem conter de UM participante ao máximo QUATRO participantes. Professores podem participar de mais de uma equipe como tutores ou orientadores.

2.3. Após a inscrição da Instituição, o responsável pode solicitar para que as equipes interessadas realizem o cadastro no site (http://www.robocodebrasil.com.br).

2.4. No cadastro, é necessário selecionar a Instituição cadastrada e informar os membros da equipe. Para os membros devem ser informados o nome e e-mail. Cada equipe deverá ter um líder.

2.5. A equipe deve se comprometer em enviar os códigos de seus robôs em todas as fases do Torneio.

3. Torneio

3.1. Rodadas

3.1.1. As rodadas começam no mês de agosto (08/08/2016). Serão cinco rodadas (uma por semana). Cada administrador do Torneio poderá escolher o dia de execução de cada rodada de acordo com a seguinte tabela:

Tabela 1: datas das rodadas

Índice	Data que ocorrerá a Rodada
1° rodada	Semana do dia 08/08/2016
2° rodada	Semana do dia 15/08/2016
3° rodada	Semana do dia 22/08/2016
4° rodada	Semana do dia 29/08/2016
5° rodada	Semana do dia 05/09/2016

A administrador poderá escolher o dia/local/hora da semana para realizar a rodada (ex. qualquer dia da semana do dia 08/08/2016 ao dia 13/08/2016 para realizar a primeira rodada).

Informar no site do Torneio Local, os seguintes dados: data, horário e local de realização da partida.

*Importante: O administrador do Torneio deve informar os seguintes dados com pelo menos dois dias de antecedência.

Exemplo:

Torneio Unicamp

Data da Rodada: **15/08/2016**

Data para o envio dos códigos de cada equipe: **14/08/2016**

Data para o administrador informar o local da partida: **13/08/2016**

3.1.2. Cada equipe será responsável por fazer upload dos códigos das classes no site, com o prazo máximo de um dia antes da execução da Rodada. Ao final de cada partida os códigos de cada equipe estarão disponíveis para acesso no site.

*Importante: Os robôs deverão ser de autoria dos participantes da equipe e devem ser totalmente desenvolvidos durante o tempo reservado de uma semana antes das rodadas seguintes. Qualquer violação destas regras, como a utilização de código pronto, trazido de casa ou baixado da internet, tanto de domínio público quanto dos protegidos por direitos autorais resultará na desqualificação da equipe.

3.1.3. As regras das rodadas se darão da seguinte maneira:

3.1.3.1. Todas as equipes se enfrentarão (todos contra todos) dentro da arena do Robocode.

3.1.3.2. Cada rodada será composta por três *rounds*, com intervalo de dez minutos entre cada round. A cada intervalo as equipes poderão alterar o código fonte dos robôs, ou ainda substituir os robôs.

3.1.3.3. No final de cada round o programa do robocode determinará um ranking com as melhores colocações que definirá as pontuações da rodada.

3.1.3.4. As pontuações de cada rodada se darão a partir da seguinte tabela:

Tabela 2: Pontuação de cada rodada

Colocação	**Pontos**
1° colocado	10
2° colocado	8
3° colocado	6
4° colocado	5
5° colocado	4
6° colocado	3
7° colocado	2
8° colocado	1
Demais	0

*Importante: equipes que não enviarem o código no prazo previsto receberão 0 pontos ao final de cada rodada.

3.1.3.5. Ao final de cada rodada, administrador do Torneio Local deve cadastrar a pontuação da rodada no site (http://www.robocodebrasil.com.br). Essa informação ficará disponível.

3.1.3.6. Ao final de cada rodada, é recomendável ao administrador do Torneio Local cadastrar os dados da rodada: fotos e/ou vídeos. Essa informação ficará disponível.

3.1.3.7. Ao final da fase de rodadas serão selecionadas quatro equipes para a fase de semifinais.

3.2. Semifinal

3.2.1. A semifinal deverá ocorrer na semana do dia 12/09/2016, e seguirá as mesmas orientações das rodadas (responsabilidades dos administradores e das equipes).

3.2.2. As quatro equipes selecionadas na fase anterior se enfrentarão individualmente de acordo com a seguinte tabela:

Tabela 3: batalhas da semifinal

1° Batalha	1° colocado	4° colocado
2° Batalha	2° colocado	3° colocado

3.2.3. Cada rodada será composta por três rounds, com intervalo de dez minutos entra cada round. A cada intervalo as equipes poderão alterar o código fonte dos robôs, ou ainda substituir os robôs.

3.2.4. O resultado dessas batalhas será fornecido pelo próprio programa do Robocode (não necessitando da tabela de pontuação).

3.2.5. Os dois campeões de cada batalha se enfrentarão na fase final. Os outros dois se enfrentarão no mesmo dia para concorrer ao título de terceiro colocado daquele torneio.

3.3. Final

3.3.1. A final deverá ocorrer na semana do dia 19/09/2016, e seguirá as mesmas orientações das rodadas (responsabilidades dos administradores e das equipes).

3.3.2. A rodada será composta por três rounds, com intervalo de dez minutos entra cada round. A cada intervalo as equipes poderão alterar o código fonte dos robôs, ou ainda substituir os robôs.

3.3.3. O resultado dessas batalhas será fornecido pelo próprio programa do Robocode (não necessitando da tabela de pontuação) e teremos o primeiro e segundo colocados de cada Torneio.

4. Liga

4.1. Classificação

4.1.1. Torneios que contenham de quatro até sete equipes classificam o primeiro colocado para a Liga. Torneios com mais de oito equipes classificam o primeiro e o segundo colocado para a Liga.

4.2. Regras

4.2.1. A Liga Nacional seguirá o mesmo formato do Torneio Local e ocorrerá no mês de setembro/2016.

4.2.2. Serão 2 rodadas (levando em consideração a mesma pontuação da Tabela 2), onde 4 equipes se classificarão para a semifinal da Liga.

4.2.3. Na semifinal, as equipes se enfrentarão no mesmo padrão da Tabela 3. Desta fase saíram os dois melhores que se enfrentarão na final e o terceiro colocado.

4.2.4. Abaixo é descrito as datas de execução das fases da Liga:

Tabela 4: datas das fases da Liga

Fase	Data
1° rodada	Semana do dia 03/10/2016
2° rodada	Semana do dia 10/10/2016
Semifinal	Semana do dia 17/10/206
Final	Semana do dia 24/10/206

4.2.5. É de responsabilidade do administrador da Liga, executar, informar as datas, lançar os dados no site, divulgar e organizar a Liga.

4.2.6. Os administradores usarão o recurso de transmissão ao vivo da mídia social YouTube para acompanhamento de todos relacionado com a Liga Nacional de Robocode.

* O ITEM 4 será administrado pela equipe do LIAG FT-UNICAMP.

5. Colaboradores

5.1. O professor interessado em colaborar com a competição, divulgando em sua região, poderá ser um colaborador da Liga. Basta que informe a administração da Liga via e-mail (contato@robocode.com.br): foto, nome, e-mail, nome da instituição de ensino e função exercida dentro da instituição.

6. Especificações Técnicas

6.1. Será criado um tutorial explicativo, para auxílio na execução de uma rodada. Neste tutorial indicaremos quais as especificações das classes e pacotes de cada robô, bem como o funcionamento do software do Robocode.

6.2. O tamanho da arena varia de acordo com o número de equipes. Torneios com até oito equipes utilizarão a proporção 800 x 800 px, torneios que possuírem acima de oito equipes, utilizarão a proporção 1200 x 1200 px.

6.3. Os responsáveis pelos torneios locais, deverão ter um computador com as especificações mínimas presentes no site oficial do Robocode.

6.4. Para os dados áudio visuais das fases dos torneios, é indicado que o responsável possua um software de captura de tela para gravar a realização das partidas, e opcionalmente

um dispositivo para fotografar os integrantes durante as fases do torneio (não há especificações quanto ao dispositivo).

7. Premiação

7.1. Torneio

7.1.1. Serão premiados os primeiros, segundos e terceiros lugares de cada Torneio, da seguinte forma:

Primeiro lugar: medalhas de ouro para todos os integrantes da equipe e troféu de primeiro lugar.

Segundo lugar: medalhas de prata para todos os integrantes da equipe.

Terceiro lugar: medalhas de bronze para todos os integrantes da equipe.

7.2. Liga

7.2.1. Serão premiados os primeiros, segundos e terceiros lugares da Liga, da seguinte forma:

Primeiro lugar: medalhas de ouro para todos os integrantes da equipe e troféu de primeiro lugar.

Segundo lugar: medalhas de prata para todos os integrantes da equipe e troféu de segundo lugar.

Terceiro lugar: medalhas de bronze para todos os integrantes da equipe e troféu de terceiro lugar.

8. Certificados

8.1. O LIAG (Laboratório de informática Aprendizado e Gestão) da FT-UNICAMP emitirá certificados de participação para todos os participantes, assim como certificados especiais para os primeiros três colocados da Liga Nacional.

V. Apêndice – Equipes Robocode Brasil 2016

Instituição	Equipes Cadastradas	Torneio	Cidade
COTIL - UNICAMP -Colégio Técnico de Limeira	Comunist Bat Galos de Batalha JMTZ T Rejecteds	Local	Limeira - SP
COTUCA - Colégio Técnico de Campinas	Faca na Caveira	Público	Campinas - SP
ETEC Adolpho Berezin	IMGF	Público	Mongaguá - SP
Faculdade Anhanguera – Campus Rio Claro	Amored Core Rocket Moto ADS Truck Hurricane Los Hermanos	Local	Rio Claro - SP
Faculdade Anhanguera – Sta Bárbara	Grupo HEMA RecrutaZero	Público	Santa Bárbara d'Oeste
Faculdade SATC	Mangilix#EngComp Nexus R2D7 Seila #EngComp SJ-001 Skynet	Local	Criciúma - SC
FT - UNICAMP Faculdade de Tecnologia	PC Computers	Público	Limeira - SP
IF Farroupilha Instituto Federal Farroupilha - Campus São Borja	OutOfTheCage 3&1/2 Dodrio Equipe Rocket KILL -9 NeonCode RoboticPampas Showtime Source Code 8bit Pumpkin	Local	São Borja - RS
IFSP – Instituo Federal São Paulo - Campus Capivari	cbl.ctrlaltdel Code Hunters Dragonborn Escudões da Sabedoria Starks The One Unnamed	Local	Capivari - SP
IFSP – Instituo Federal São Paulo - Campus Hortolândia	Genesis IFSP Mec Team Baidu	Público	Hortolândia - SP
PUC - Pontifícia Universidade Católica do Paraná	Dot GTX Equipe Dois Ômega TStark	Local	Londrina - PR
UFBA -Universidade Federal da Bahia	BIRL - Beyond Intelligent Robotic Lifeform Liga da Justiça Orion Skynet UFBA WeDontHaveName	Local	Salvador - BA
UFMG -Universidade Federal de Minas Gerais	Der Mechanische Shatten	Público	Belo Horizonte - MG
UNICAMP - Universidade Estadual de Campinas	GER	Público	Campinas - SP

VI. Apêndice – Questionário Inicial

Robocode Brasil – Questionário Inicial
Projeto de pesquisa UNICAMP cadastrado e aprovado na Plataforma Brasil pelo Comitê de Ética em Pesquisa (CEP). Para maiores informações acessar: http://goo.gl/00reIa, em busca de pesquisas aprovadas, buscar a palavra chave "Robocode" ou confirmar aprovação pelo CEP. Número do CAAE: 53624816.3.0000.5404, Número do Perecer: 1499745
*Obrigatório

1. Qual sua Instituição? *

2. Qual curso e período? *

3. Gênero: *
- ✓ Masculino
- ✓ Feminino
- ✓ Outro:

4. Qual sua idade? *
- ✓ Menos de 15 anos
- ✓ Entre 15 e 18 anos
- ✓ Entre 19 e 22 anos
- ✓ Entre 23 e 26 anos
- ✓ Entre 27 e 30 anos
- ✓ Entre 31 e 35 anos
- ✓ Entre 35 e 49 anos
- ✓ Acima de 50 anos

5. Renda familiar:
- ✓ Até 3 Salários Mínimos (R$ 2640,00)
- ✓ de 3 até 5 Salários Mínimos (R$ 2640,00 - R$ 4400,00)
- ✓ de 5 até 8 Salários Mínimos (R$ 4400,00 - R$ 7040,00)
- ✓ Mais que 8 Salários Mínimos (Acima de R$ 7040,00)
- ✓ Não desejo responder

6. Como você classifica disciplinas que envolvem linguagem de programação de modo geral em relação ao grau de dificuldade? *
- ✓ Muito Fácil
- ✓ Fácil
- ✓ Médio
- ✓ Difícil
- ✓ Muito Difícil

Antes de continuar, veja alguns dos tipos de aulas

AULA EXPOSITIVA CLÁSSICA. É a mais comum nas escolas de todos os níveis. Nesse tipo de aula, o professor fala sobre determinado assunto durante algum tempo, sendo que a postura dos alunos é predominantemente passiva.

AULA DIALOGADA (DIALÓGICA). O professor tenta quebrar a postura passiva dos seus alunos, por meio da introdução de questionamentos a serem respondidos pelos alunos, dinamizando a atividade em sala de aula.

AULA DE DEMONSTRAÇÃO. Um tipo de aula em que o professor, com o uso de equipamentos e outros materiais, inclusive experimentais, demonstra a operação, os efeitos ou mesmo uma lei científica.

AULA PRÁTICA. Uso de equipamentos e materiais, com os quais os alunos fazem algum tipo de experiência sobre uma lei científica ou os efeitos dela, relacionando seus aspectos teóricos e práticos, sendo, por isso, uma metodologia de trabalho ativa.

AULAS DIVERSIFICADAS. Uso de Problemas reais, Jogos, grupos de pesquisa, debates, utilização de tecnologias atuais no auxílio do aprendizado. Alunos com postura em buscar conhecimento mediados por professores.

Antes de prosseguir, analise também a seguinte imagem.

Aula Expositiva Clássica

Aula Diversificada

7. Ainda com relação as disciplinas que envolvem linguagem de programação, selecione o(s) tipo(s) de aula(s) predominantemente utilizado(s) em sua Instituição?
Muito Utilizada

	Muito Utilizada	Utilizada	Pouco Utilizada	Raramente	Nunca
AULA EXPOSITIVA CLÁSSICA	○	○	○	○	○
AULA DIALOGADA (DIALÓGICA)	○	○	○	○	○
AULA DE DEMONSTRAÇÃO	○	○	○	○	○
AULA PRÁTICA	○	○	○	○	○
AULAS DIVERSIFICADAS	○	○	○	○	○

8. O que motivou sua participação na competição científica de programação Robocode Brasil? *

É possível selecionar várias respostas igualmente válidas.

- ✓ Jogar
- ✓ Participar de uma equipe
- ✓ Representar a Instituição
- ✓ Competir
- ✓ Aprender
- ✓ Professores
- ✓ Colegas
- ✓ Entreterimento
- ✓ Premiação
- ✓ Experimentar novas formas de aprender programação
- ✓ Outro:

9. De que modo se preparou para o torneio? Quais fontes utilizou para preparação? *

É possível selecionar várias respostas igualmente válidas.

- ✓ Auxílio de professores / tutores / colegas
- ✓ Estudo em grupo
- ✓ Estudo em sala de aula
- ✓ Estudo em casa
- ✓ Análise de torneios anteriores
- ✓ Análise de outros torneios no país / mundo
- ✓ Mini curso disponível no http://www.robocodebrasil.com.br/mini-curso.php
- ✓ Outros materiais disponíveis na internet
- ✓ Outro:

10. Qual(quais) disciplina(s) que cursa ou cursou considera que auxiliou na preparação para torneio? *

É possível selecionar várias respostas igualmente válidas.

- ✓ Algoritmos de Programação
- ✓ Lógica de Programação
- ✓ Linguagem de Programação de Computadores (incluindo web)
- ✓ Programação Orientada a Objeto
- ✓ Inteligência Artificial
- ✓ Robótica
- ✓ Matemática, Cálculo e Álgebra

- ✓ Projetos e Gestão
- ✓ Outro:

11. Participou de outras iniciativas com envolvimento de jogos digitais dentro da sua Instituição? Se sim, qual (quais)? *

12. Conhece outros jogos digitais que auxiliam no ensino de programação de computadores? Se sim, qual (quais)? *

13. Contato e-mail:
Seu contato de e-mail é muito importante. Ao final da competição receberá um questionário para dar seu feedback em relação ao Robocode Brasil

VII. Apêndice – Questionário Final

Robocode Brasil - Final
Projeto de pesquisa UNICAMP cadastrado e aprovado na Plataforma Brasil pelo Comitê de Ética em Pesquisa (CEP). Para maiores informações acessar: http://goo.gl/00reIa, em busca de pesquisas aprovadas, buscar a palavra chave "Robocode" ou confirmar aprovação pelo CEP. Número do CAAE: 53624816.3.0000.5404, Número do Perecer: 1499745

*Obrigatório

Antes de prosseguir, analise as figuras com relação a dois tipos distintos de aulas: (1) aula expositiva clássica e (2) aula diversificada.

1

Aula Expositiva Clássica

2

Aula Diversificada

1. Em uma Aula Expositiva Clássica (figura 1) de 60 minutos da disciplina de linguagem de programação, quanto tempo você considera que consegue prestar atenção? *
Aula de 60 minutos com apresentação de conceitos como: sintaxe; variáveis; estruturas condicionais; estrutura de repetição.

- ✓ Entre 1 a 10 minutos
- ✓ Entre 11 a 20 minutos
- ✓ Entre 21 a 30 minutos
- ✓ Entre 31 a 40 minutos
- ✓ Entre 41 a 50 minutos
- ✓ Entre 51 a 60 minutos

2. Em uma Aula Diversificada com envolvimento de equipes e jogos digitais (figura 2) de 60 minutos da disciplina de linguagem de programação, quanto tempo você considera que consegue prestar atenção? *
Aula de 60 minutos com apresentação de conceitos como: sintaxe; variáveis; estruturas condicionais; estrutura de repetição.

- ✓ Entre 1 a 10 minutos
- ✓ Entre 11 a 20 minutos
- ✓ Entre 21 a 30 minutos
- ✓ Entre 31 a 40 minutos
- ✓ Entre 41 a 50 minutos
- ✓ Entre 51 a 60 minutos

3. Entre: Expositiva Clássica (figura 1) e Aula Diversificada com Jogos (figura 2), qual a sua preferência se pudesse escolher seus tipos de aulas. *

- ✓ Expositiva Clássica (figura 1)
- ✓ Predominantemente Expositiva Clássica (figura 1)
- ✓ Misto entre Expositiva Clássica (figura 1) e Aula Diversificada (figura 2)
- ✓ Predominantemente Aula Diversificada (figura 2)
- ✓ Aula Diversificada (figura 2)

4. Considera que o torneio Robocode auxiliou e/ou melhorou o entendimento de conceitos e práticas de linguagem de programação? *

- ✓ Sim
- ✓ Não
- ✓ Outro:

5. Desafios como melhorar a movimentação do Robô (ex. método run() ou método onHitWall()) são encontrados durante a competição. Em uma escala de 0 a 5 quanto você acha que evoluiu os métodos de movimentação do seu Robô da primeira a última etapa disputada? *
- Pouca Evolução

- ✓ 1
- ✓ 2
- ✓ 3
- ✓ 4
- ✓ 5

- Muita Evolução

6. Quais características você e sua equipe adotaram para resolver os desafios encontrados no jogo durante a competição? *

	Sempre	Frequentemente	Moderadamente	Raramente	Nunca
Criar estratégias conforme os problemas enfrentados no jogo	○	○	○	○	○
Formular objetivos de acordo com os métodos do Robô (movimentação, disparo, energia etc.)	○	○	○	○	○
Promover debates e discussões em grupo a fim de resolver problemas do Robô	○	○	○	○	○
Rediscutiram problemas com base nos novos conhecimento adquiridos	○	○	○	○	○

7. Se o Robocode auxiliou e/ou melhorou a compreensão, quais conceitos considera que houve melhoria? *
É possível selecionar várias respostas igualmente válidas.

- ✓ Sintaxe
- ✓ Variáveis
- ✓ Estruturas condicionais
- ✓ Estrutura de repetição.
- ✓ Não houve nenhuma melhoria
- ✓ Outro:

8. Considera que os jogos digitais (ex. Robocode e outros) poderiam ser ferramentas utilizadas nas aulas presenciais para auxílio na compreensão de conceitos e práticas? *

- ✓ Sim
- ✓ Não
- ✓ Outro:

9. SE pudesse escolher, gostaria de estudar conceitos e práticas de programação em aulas clássicas OU em diversificação de metodologias que envolvessem o ensino-aprendizagem com base em jogos? *

- ✓ Aulas Clássicas
- ✓ Aulas Clássicas E Aulas diversificadas com jogos
- ✓ Aulas diversificadas com jogos
- ✓ Outro:

10. Considera motivador programar em ferramentas como o Robocode dentro de competições? Se sim, o que ela pode trazer de diferencial em relação ao ensino tradicional (aula clássica)? *
É possível selecionar várias respostas igualmente válidas.

- ✓ Ambiente divertido para desenvolver as competências presentes nas ementas das disciplinas de linguagem de programação
- ✓ O compromisso com o torneio em acompanhar os resultados possibilitou criar um senso crítico na tentativa de constante melhoria dos programas e da estratégia de jogo.

- ✓ Geração de interações entre as equipes participantes, colegas e professores
- ✓ Situações encontradas no jogo encoraja a pesquisar de forma criativa e exploratória
- ✓ Aprofundar a pesquisa de conceitos abordados em disciplinas de linguagem de programação
- ✓ Trabalhar em equipes para desenvolver estratégias para o Robô
- ✓ Competição estimula o interesse em aperfeiçoar o código com objetivo de vencer
- ✓ Estudar conceitos ainda não vistos nas aulas
- ✓ Despertar a curiosidade intelectual
- ✓ Melhoria no processo de aprendizagem
- ✓ Ter a oportunidade de analisar os códigos dos adversários permite as equipes diversificar as estratégias a cada disputa.
- ✓ Conferir o os resultados e a evolução durante a competição
- ✓ Auxilia o entendimento de conceitos e práticas
- ✓ Ferramenta complementar ao estudo de linguagem de programação
- ✓ Oferece nivelamento entre equipes mais experientes e iniciantes.
- ✓ Outro:

11. Analise a Imagem para Avaliar a Motivação Satisfação com Torneio

Avaliação da Satisfação *

	1	2	3	4	5
Satisfação geral no Robocode Brasil	○	○	○	○	○
Satisfação na sua Instituição	○	○	○	○	○
Satisfação na sua Equipe	○	○	○	○	○
Resultados da sua equipe	○	○	○	○	○
Envolvimento da sua Instituição	○	○	○	○	○

VIII. Apêndice – Parecer Substanciado CEP

COMITÊ DE ÉTICA EM PESQUISA DA UNICAMP - CAMPUS CAMPINAS

PARECER CONSUBSTANCIADO DO CEP

DADOS DO PROJETO DE PESQUISA

Título da Pesquisa: APRENDIZAGEM BASEADA EM PROBLEMAS APLICADA AO ENSINO DE LINGUAGEM DE PROGRAMAÇÃO COM ROBOCODE

Pesquisador: Matheus Carvalho Meira

Área Temática:

Versão: 2

CAAE: 53624816.3.0000.5404

Instituição Proponente: Faculdade de Tecnologia

Patrocinador Principal: Financiamento Próprio

DADOS DO PARECER

Número do Parecer: 1.499.745

Apresentação do Projeto:

A grande retenção dos alunos na disciplina de linguagem de programação do primeiro ano dos cursos técnicos na área de informática, representa um importante tema discussão entre professores e gestores de escolas técnicas. Este projeto busca estudar métodos de aprendizagem para avaliação e comparação da eficácia entre a metodologia de aprendizado baseado em problemas (PBL) e o Ensino Tradicional. É proposto o desenvolvimento de um ambiente para aplicação da proposta curricular do PBL no ensino de linguagem de programação com jogo educacional digital em cursos técnicos de informática.

A presente proposta de pesquisa busca a sustentação teórica nos conceitos da metodologia de aprendizado PBL associado a software, do tipo jogo educacional, com o Robocode. O estudo de caso será a liga Robocode/FT-UNICAMP. O objetivo do estudo de caso é identificar as potencialidades que o ambiente de ensino-aprendizagem pode proporcionar nas disputas entre equipes envolvendo atividades com linguagem de programação. Com base no estudo de caso, serão feitas análises quantitativas com aplicação de questionários e qualitativas com mapas conceituais com julgamento de banca técnica.

Endereço: Rua Tessália Vieira de Camargo, 126
Bairro: Barão Geraldo **CEP:** 13.083-887
UF: SP **Município:** CAMPINAS
Telefone: (19)3521-8936 **Fax:** (19)3521-7187 **E-mail:** cep@fcm.unicamp.br

Objetivo da Pesquisa:

Objetivo Primário:

Avaliar se uma proposta curricular com jogo educacional/framework Robocode com base na PBL estimula a aprendizagem em disciplinas de linguagem de programação nos cursos técnicos de informática.

Objetivo Secundário:

(1) Propor um problema na disciplina de linguagem de programação associado ao Robocode; (2) Implantar PBL em uma disciplina do currículo de cursos técnicos na área de Informática; (3) Permitir aos alunos desenvolverem a construção do conhecimento, de acordo com currículo da disciplina, com base na solução de problemas em um ambiente gerenciado; (4) Avaliar se a metodologia de aprendizagem PBL aliada ao Robocode interfere no desempenho do processo avaliativo; (5) Analisar a eficácia do uso de PBL/Jogos educacionais; (6) Estudar o Torneio Robocode.

Avaliação dos Riscos e Benefícios:

Riscos:

Ambiente livre de riscos e com feedback imediato. Alunos são estimulados a explorar, experimentar, cometer erros e tentar novamente.

Benefícios:

Benefícios à aprendizagem pelo fato de fornecer um feedback imediato. As instituições de ensino podem obter benefícios com os avanços tecnológicos dos softwares educacionais para auxiliar o processo de ensino-aprendizagem. Para isso, podem utilizar softwares educacionais, do tipo jogos, combinados com métodos de ensino para desenvolver ambientes de aprendizagem em disciplinas de linguagem de programação.

Comentários e Considerações sobre a Pesquisa:

Trata-se de um projeto de mestrado da FT-UNICAMP em conjunto com o Instituto Federal de Educação Ciência e Tecnologia de São Paulo (IFSP) do Campus Capivari, onde será realizada a pesquisa com alunos do segundo e terceiro ano do curso Técnico Integrado em Informática. A pesquisa visa realizar um estudo sobre aprendizagem baseada em problema utilizando o Robocode. O pesquisador utilizará questionários (N=20) a serem aplicados aos alunos.

A pesquisa é pertinente e embasada na literatura. Não há riscos previsíveis e um possível benefício

direto aos participantes da pesquisa será a melhora do aprendizado.

Considerações sobre os Termos de apresentação obrigatória:

Todos os documentos foram apresentados:

1) Folha de rosto, assinada pelo diretor da FT-UNICAMP e carta de autorização do Instituto Federal de Educação Ciência e Tecnologia de São Paulo, devidamente assinados.

2) Projeto de Pesquisa gerado pela Plataforma Brasil, com o cronograma e orçamento adequados.

3) Projeto de pesquisa detalhado e o questionário, devidamente escrito e referenciado

4) TCLE, devidamente redigido.

Recomendações:

Conclusões ou Pendências e Lista de Inadequações:

Uma vez que o projeto é pertinente e embasado na literatura, todos os termos estão de acordo com as regras do CEP - CONEP e todas as pendências foram atendidas, recomento a aprovação.

Lista de Pendências da Primeira Avaliação:

Resposta: resposta foi introduzida no sistema da Plataforma Brasil no item 5 "Outras Informações" no Campo de Orçamento Financeiro em Detalhamento do Orçamento: Detalhamento do Orçamento inserido: Material de Expediente, Custeio R$250,00; Material de Processamento de Dados, Custeio R$150,00; Serviços Gráficos, Custeio R$80,00; Transportes para desenvolvimento da Pesquisa, Custeio R$200,00; Total de R$680,00.

Situação: Os benefícios foram corrigidos.

Conclusão: Pendência atendida.

2) Projeto de Pesquisa gerado pela Plataforma Brasil (PB_INFORMAÇÕES_BÁSICAS_DO_PROJETO_655487.pdf):

O orçamento está com custo zero. É difícil ter um orçamento igual a zero, onde nem material de escritório vai ser gasto. Portanto, readequar e detalhar o orçamento ou justificar o custo nulo.

Resposta: resposta foi introduzida no sistema da Plataforma Brasil no item 5 "Outras Informações" no Campo de Orçamento Financeiro em Detalhamento do Orçamento: Detalhamento do Orçamento inserido: Material de Expediente, Custeio R$250,00; Material de Processamento de Dados, Custeio

R$150,00; Serviços Gráficos, Custeio R$80,00; Transportes para desenvolvimento da Pesquisa, Custeio R$200,00; Total de R$680,00.
Situação: o orçamento foi inserido.
Conclusão: Pendência atendida.

3) TCLE:
2.1- Deixar claro que não existem riscos previsíveis e qual é o possível benefício que tem que ser direto ao participante da pesquisa. Readequar.
Resposta:
Riscos:
Não existe nenhum tipo de risco ao aluno participante, toda ementa da disciplina será apresentada. Todos conceitos teóricos e práticos que regem a disciplina de linguagem de programação no plano de aula e ensino, documentos pertencentes ao plano do curso Técnico Integrado ao Ensino Médio do IFSP (aprovado pelo MEC), serão abortados em sua íntegra.
Benefícios:
Aprender os conceitos da disciplina de linguagem de programação de modo lúdico com diversificação de métodos de ensino. Proporcionar grupos colaborativos para estudo de linguagem de programação. Participar de ambientes com jogos educacionais digitais no auxílio do ensino-aprendizagem. O aluno atua como coautor de seu aprendizado. Feedback estruturado no jogo permite analisar a compreensão dos conceitos abordados. Uma vez compreendido, as situações planejadas no jogo possibilitam a evolução planejada dos estudos.
Situação: Os riscos/benefícios foram corrigidos.
Conclusão: Pendência atendida.

2.2- Inserir o contato do orientador: nome, endereço profissional, telefone, email ou outra forma de contato. É importante lembrar que o endereço profissional deverá incluir o departamento e/ou ambulatório de atuação dos pesquisadores, para que sejam prontamente localizados. Readequar.
Resposta:
Orientador da pesquisa: Dr. Marcos Augusto Francisco Borges (Professor Pleno UNICAMP), coordenador do laboratório de LIAG (laboratório de Informática, Aprendizagem e Gestão) da FT-UNICAMP (Faculdade de Tecnologia – FT - UNICAMP - Universidade Estadual de Campinas), localizado na R. Paschoal Marmo, 1888 - CEP:13484-332 - Jd. Nova Itália - Limeira, SP. O LIAG está localizado na sala 15 FT-UNICAMP no prédio administrativo no corredor onde se

encontra a secretária e sala dos professores do lado do CAT. Telefone: (19) 2113-3332, as terçasfeiras das 13h00 às 22h30 no LIAG. Contato e-mail: marcosborges@ft.unicamp.br.
Situação: foi inserido o contato do orientador.
Conclusão: Pendência atendida.

2.3- Em justificativas e objetivos, deixar claro que o segundo parágrafo é o objetivo e explicar o que é PLB. Readequar.
Resposta:
Justificativa:
Não se pode se fixar apenas em meios tradicionais de ensino. É importante combinar diferentes métodos para atrair e incentivar os alunos a conquistar os objetivos propostos no currículo da disciplina. As instituições de ensino podem obter benefícios com os avanços tecnológicos dos softwares educacionais para auxiliar o processo de ensino-aprendizagem. Para isso, podem utilizar softwares educacionais, do tipo jogos, combinados com métodos de ensino para desenvolver ambientes de aprendizagem em disciplinas de linguagem de programação.
Objetivos:
Avaliar se uma proposta curricular com jogo educacional/framework Robocode com base na metodologia de aprendizado baseado em problemas (Problem Based Learning – PBL) estimula a aprendizagem em disciplinas de linguagem de programação nos cursos técnicos de informática.
Situação: o TCLE foi modificado.
Conclusão: Pendência atendida.

2.4- Em Consentimento livre e esclarecido, por se tratar de uma população vulnerável (menores de idade), manter o item com os dados e assinatura do responsável legal e colocar um assentimento para o menor de idade, dizendo que concorda ele em participar da pesquisa com o lugar para nome e assinatura.
Resposta:
Consentimento livre e esclarecido:
Após ter sido esclarecimento sobre a natureza da pesquisa, seus objetivos, métodos, desconfortos e riscos, benefícios previstos, divulgação de imagem e projetos criados, aceito participar:
Nome do(a) participante: _____

RG: _____ CPF: _____

E-mail: _____ Telefone: _____

_____ Data: ____/_____/_____.

(Assinatura do participante ou nome e assinatura do seu responsável LEGAL)

Responsabilidade do Pesquisador:

Asseguro ter cumprido as exigências da resolução 466/2012 CNS/MS e complementares na elaboração do protocolo e na obtenção deste Termo de Consentimento Livre e Esclarecido.

Asseguro, também, ter explicado e fornecido uma cópia deste documento ao participante.

Informo que o estudo foi aprovado pelo CEP perante o qual o projeto foi apresentado e pela CONEP, quando pertinente. Comprometo-me a utilizar o material e os dados obtidos nesta pesquisa exclusivamente para as finalidades previstas neste documento ou conforme o consentimento dado pelo participante.

_____ Data: ____/_____/_____.

(Assinatura do pesquisador)

Situação: o TCLE foi modificado.

Conclusão: Pendência atendida.

Considerações Finais a critério do CEP:

- O sujeito de pesquisa deve receber uma via do Termo de Consentimento Livre e Esclarecido, na íntegra, por ele assinado (quando aplicável).

- O sujeito da pesquisa tem a liberdade de recusar-se a participar ou de retirar seu consentimento em qualquer fase da pesquisa, sem penalização alguma e sem prejuízo ao seu cuidado (quando aplicável).

- O pesquisador deve desenvolver a pesquisa conforme delineada no protocolo aprovado. Se o pesquisador considerar a descontinuação do estudo, esta deve ser justificada e somente ser realizada após análise das razões da descontinuidade pelo CEP que o aprovou. O pesquisador deve aguardar o parecer do CEP quanto à descontinuação, exceto quando perceber risco ou dano não previsto ao sujeito participante ou quando constatar a superioridade de uma estratégia diagnóstica ou terapêutica oferecida a um dos grupos da pesquisa, isto é, somente em caso de necessidade de ação imediata com intuito de proteger os participantes.

- O CEP deve ser informado de todos os efeitos adversos ou fatos relevantes que alterem o curso normal do estudo. É papel do pesquisador assegurar medidas imediatas adequadas frente a evento adverso grave ocorrido (mesmo que tenha sido em outro centro) e enviar notificação ao CEP e à Agência Nacional de Vigilância Sanitária – ANVISA – junto com seu posicionamento.

- Eventuais modificações ou emendas ao protocolo devem ser apresentadas ao CEP de forma clara e sucinta, identificando a parte do protocolo a ser modificada e suas justificativas e aguardando a aprovação do CEP para continuidade da pesquisa. Em caso de projetos do Grupo I ou II apresentados anteriormente à ANVISA, o pesquisador ou patrocinador deve envia-las também à mesma, junto com o parecer aprovatório do CEP, para serem juntadas ao protocolo inicial.

- Relatórios parciais e final devem ser apresentados ao CEP, inicialmente seis meses após a data deste parecer de aprovação e ao término do estudo.

-Lembramos que segundo a Resolução 466/2012 , item XI.2 letra c, "cabe ao pesquisador apresentar dados solicitados pelo CEP ou pela CONEP a qualquer momento".

Este parecer foi elaborado baseado nos documentos abaixo relacionados:

Tipo Documento	Arquivo	Postagem	Autor	Situaçao
Informações Basicas do Projeto	PB_INFORMAÇÕES_BÁSICAS_DO_PROJETO_655487.pdf	04/04/2016 18:37:56		Aceito
Outros	CartaResposta.pdf	04/04/2016 18:35:26	Matheus Carvalho Meira	Aceito
TCLE / Termos de Assentimento / Justificativa de Ausência	TCLETermoDeConsentimentoLivreEEsclarecidoCorecao.pdf	04/04/2016 18:33:47	Matheus Carvalho Meira	Aceito
Cronograma	Cronograma.pdf	17/02/2016 09:46:58	Matheus Carvalho Meira	Aceito
Projeto Detalhado / Brochura Investigador	ProjetoDePesquisa.pdf	17/02/2016 09:46:11	Matheus Carvalho Meira	Aceito
Outros	ComprovacaoDeVinculoIFSP.pdf	17/02/2016 09:44:38	Matheus Carvalho Meira	Aceito

Endereço: Rua Tessália Vieira de Camargo, 126
Bairro: Barão Geraldo **CEP:** 13.083-887
UF: SP **Município:** CAMPINAS
Telefone: (19)3521-8936 **Fax:** (19)3521-7187 **E-mail:** cep@fcm.unicamp.br

Outros	AtestadoMatriculaID.pdf	17/02/2016 09:43:17	Matheus Carvalho Meira	Aceito
Folha de Rosto	FolhaDeRosto.pdf	17/02/2016 09:40:06	Matheus Carvalho Meira	Aceito
Outros	AtestadoMatricula.pdf	15/02/2016 10:41:46	Matheus Carvalho Meira	Aceito
Outros	QuestoesDaEntrevista.pdf	09/02/2016 17:15:56	Matheus Carvalho Meira	Aceito
Outros	AutorizacaoDaInstituicaoCoparticipanteIFSP.pdf	09/02/2016 17:14:46	Matheus Carvalho Meira	Aceito

Situação do Parecer:
Aprovado

Necessita Apreciação da CONEP:
Não

CAMPINAS, 14 de Abril de 2016

Assinado por:
Renata Maria dos Santos Celeghini
(Coordenador)

IX. Apêndice – Termo de Consentimento TLCE

TERMO DE CONSENTIMENTO LIVRE E ESCLARECIDO

APRENDIZAGEM BASEADA EM PROBLEMAS APLICADA AO ENSINO DE LINGUAGEM DE PROGRAMAÇÃO COM ROBOCODE

Matheus Carvalho Meira

Número do CAAE: 53624816.3.0000.5404

Você está sendo convidado a participar como voluntário de um estudo. Este documento, chamado Termo de Consentimento Livre e Esclarecido, visa assegurar seus direitos como participante e é elaborado em duas vias, uma que deverá ficar com você e outra com o pesquisador.

Por favor, leia com atenção e calma, aproveitando para esclarecer suas dúvidas. Se houver perguntas antes ou mesmo depois de assiná-lo, você poderá esclarecê-las com o pesquisador. Se preferir, pode levar para casa e consultar seus familiares ou outras pessoas antes de decidir participar. Se você não quiser participar ou retirar sua autorização, a qualquer momento, não haverá nenhum tipo de penalização ou prejuízo.

Justificativa:

Não se pode se fixar apenas em meios tradicionais de ensino. É importante combinar diferentes métodos para atrair e incentivar os alunos a conquistar os objetivos propostos no currículo da disciplina. As instituições de ensino podem obter benefícios com os avanços tecnológicos dos softwares educacionais para auxiliar o processo de ensino-aprendizagem. Para isso, podem utilizar softwares educacionais, do tipo jogos, combinados com métodos de ensino para desenvolver ambientes de aprendizagem em disciplinas de linguagem de programação.

Objetivos:

Avaliar se uma proposta curricular com jogo educacional/framework Robocode com base na metodologia de aprendizado baseado em problemas (*Problem Based Learning* – PBL) estimula a aprendizagem em disciplinas de linguagem de programação nos cursos técnicos de informática.

Procedimentos:

No estudo, você está sendo convidado a: participar de um projeto no IFSP (Campus Capivari) em uma parceria com FT-UNICAMP (laboratório de Informática, Aprendizagem e Gestão - LIAG) com a utilização do jogo educacional digital Robocode associado a método de aprendizagem baseado em problemas para estudo de conceitos relacionados a disciplina de Linguagem de Programação de modo lúdico. Durante as atividades, como forma de registro do projeto, haverá entrevistas e/ou questionários durante a apresentação dos conceitos da disciplina, além do projeto e programação de robôs virtuais em laboratórios de informática da instituição.

As atividades serão feitas dentro do semestre letivo de aula utilizando 8 dias de aula em semanas seguidas. Cada aula terá duração de 50 minutos e ocorrerá em dias e horários de sua aula da disciplina de Linguagem de Programação e/ou em horários previstos para atividades do curso Técnico Integrado em Informática do IFSP Campus Capivari.

Os processos do curso de linguagem de programação com aprendizagem baseada em problemas associadas ao jogo tem por objetivo ensinar conceitos da disciplina. Ao final da apresentação dos conceitos, como forma de aplicação prática para resolução de problemas entre equipes, haverá um torneio local de programação de robôs no jogo Robocode. Durante o torneio serão estudados problemas reais que envolve a programação na área de informática, para que cada equipe possa levantar hipóteses, criar objetivos junto a programação e definir uma estratégia de resolução.

Rubrica do pesquisador:_____ Rubrica do participante:_____

Não haverá nenhum tipo de ressarcimento de despesas ao aluno como, por exemplo, referente à alimentação ou transporte. As atividades serão realizadas dentro dos horários da disciplina de Linguagem de programação ou reservada atividades do curso.

Riscos:

Não existe nenhum tipo de risco ao aluno participante, toda ementa da disciplina será apresentada. Todos conceitos teóricos e práticos que regem a disciplina de linguagem de programação no plano de aula e ensino, documentos pertencentes ao plano do curso Técnico Integrado ao Ensino Médio do IFSP (aprovado pelo MEC), serão abortados em sua íntegra.

Benefícios:

Aprender os conceitos da disciplina de linguagem de programação de modo lúdico com diversificação de métodos de ensino. Proporcionar grupos colaborativos para estudo de linguagem de programação. Participar de ambientes com jogos educacionais digitais no auxílio do ensino-aprendizagem. O aluno atua como coautor de seu aprendizado. Feedback estruturado no jogo permite analisar a compreensão dos conceitos abordados. Uma vez compreendido, as situações planejadas no jogo possibilitam a evolução planejada dos estudos.

Problemas relacionados aos torneios devem despertar o interesse dos alunos na pesquisa pela informação de modo a criar discussões, levantar hipóteses e gerar perspectivas de resolução. Um curso que ensine linguagem de programação e prepare alunos para disputar torneios deverá estimular a aprendizagem com práticas no Robocode. A proposta visa proporcionar sugestões de melhoria do processo de ensino-aprendizagem e da pesquisa dos conceitos abordados na disciplina de linguagem de programação para os cursos técnicos na área de informática.

Sigilo e privacidade:

Fotos de algumas aulas e os projetos criados pelos alunos poderão vir a ser divulgados no Documento de Pesquisa. Todas as imagens serão cuidadosamente alteradas, com a colocação de tarja preta na parte frontal do rosto na área dos olhos, para que os alunos não sejam identificados.

Você tem a garantia de que sua identidade será mantida em sigilo e nenhuma informação pessoal será dada a outras pessoas que não façam parte da equipe de pesquisadores. Na divulgação dos resultados desse estudo, seu nome não será citado.

Contato:

Em caso de dúvidas sobre o estudo, você poderá entrar em contato com a pesquisador Matheus Carvalho Meira, no IFSP Instituto Federal de Educação, Ciência e Tecnologia de São Paulo Campus Capivari, Avenida Doutor Ênio Pires de Camargo, 2971 - São João Batista - Capivari SP- CEP: 13360-000 - Telefone: (19) 3492-8585, as terças-feiras das 19h00 às 22h15 nas salas dos professores e/ou dependências do prédio dos laboratórios de informática. Contato e-mail: meira@ifsp.edu.br ou m108479@dac.unicamp.br.

Orientador da pesquisa: Dr. Marcos Augusto Francisco Borges (Professor Pleno UNICAMP), coordenador do laboratório de LIAG (laboratório de Informática, Aprendizagem e Gestão) da FT-UNICAMP (Faculdade de Tecnologia – FT - UNICAMP - Universidade Estadual de Campinas), localizado na R. Paschoal Marmo, 1888 - CEP:13484-332 - Jd. Nova Itália - Limeira, SP. O LIAG está localizado na sala 15 FT-UNICAMP no prédio administrativo no corredor onde se encontra a secretária e sala dos professores do lado do CAT. Telefone: (19) 2113-3332, as terças-feiras das 13h00 às 22h30 no LIAG. Contato e-mail: marcosborges@ft.unicamp.br.

Em caso de denúncias ou reclamações sobre sua participação e sobre questões éticas do estudo, você pode entrar em contato com a secretaria do Comitê de Ética em Pesquisa (CEP) da UNICAMP: Rua: Tessália Vieira de Camargo, 126; CEP 13083-887 Campinas – SP; telefone (19) 3521-8936; fax (19) 3521-7187; e-mail: cep@fcm.unicamp.br

Rubrica do pesquisador:_____ Rubrica do participante:_____

Consentimento livre e esclarecido:

Após ter sido esclarecimento sobre a natureza da pesquisa, seus objetivos, métodos, desconfortos e riscos, benefícios previstos, divulgação de imagem e projetos criados, aceito participar:

Nome do(a) participante: _____

RG: _____ CPF: _____

E-mail: _____ Telefone: _____

_____ Data: ____/_____/_____.
(Assinatura do participante ou nome e assinatura do seu responsável LEGAL)

Responsabilidade do Pesquisador:

Asseguro ter cumprido as exigências da resolução 466/2012 CNS/MS e complementares na elaboração do protocolo e na obtenção deste Termo de Consentimento Livre e Esclarecido. Asseguro, também, ter explicado e fornecido uma cópia deste documento ao participante. Informo que o estudo foi aprovado pelo CEP perante o qual o projeto foi apresentado e pela CONEP, quando pertinente. Comprometo-me a utilizar o material e os dados obtidos nesta pesquisa exclusivamente para as finalidades previstas neste documento ou conforme o consentimento dado pelo participante.

_____ Data: ____/_____/_____.
(Assinatura do pesquisador)

DECLARAÇÃO DO RESPONSÁVEL LEGAL (EM CASO DE MENOR DE IDADE)

Nome do responsável: _____

RG: _____ CPF: _____

E-mail: _____ Telefone: _____

DECLARAÇÃO DO RESPONSÁVEL
Declaro para os devidos fins, que sou o responsável legal do(a) menor supracitado(a), e que estou ciente e de acordo com participação voluntária do presente estudo. Declaro, ainda, que autorizo o mesmo a participar como voluntário(a) do estudo do presente projeto de pesquisa.

_____ Data: ____/_____/_____.
(Assinatura do responsável LEGAL)

Rubrica do pesquisador:_____ Rubrica do participante:_____

ANEXOS

I. Anexo – Instituições com Incidência de Competições Robocode.

Nº	Instituição / Cidade / Ano(s) da incidência(s) / Link de Registro
1	Colégio São Luis - Dores do Indaiá-MG (2012): http://www.konkuri.com/tournaments/04a5c65a91
2	COTIL-UNICAMP - Colégio Técnico de Limeira - Limeira-SP (2015, 2014, 2013, 2012, 2011): http://torneiorobocode.orgfree.com/torneio-cotil.php
3	Estácio - Centro Universitário UniSEB - Ribeirão Preto-SP (2010): http://ccpecp.blogspot.com.br/2010/11/rocode-na-disciplina-de-robotica-da-eng.html
4	ETEC Adolpho Berezin - Mongaguá-SP (2015): http://eteab.com.br/cms/index.php/2015/10/02/
5	ETEC Centro Paula Souza - Armando Pannunzio - Sorocaba-SP (2015): http://www.jornalcruzeiro.com.br/tvcruzeiro/000001023
6	ETEC Centro Paula Souza - Dr. José Luiz Viana Coutinho - Jales-SP (2014): http://tecnoinfojales.blogspot.com.br/2014/09/torneio-de-robocode.html
7	ETEC Centro Paula Souza - Fernando Prestes Sorocaba-SP (2015): http://www.etecfernandoprestes.com.br/noticia/104/campeonato-robocode
8	ETEC Centro Paula Souza - Jão Belarmino - Amparo-SP - (2014, 2013, 2012, 2011): http://www.etecjoaobelarmino.com.br/informatica_robocode.html
9	ETEC Centro Paula Souza - Lauro Gomes - São Bernardo do Campo-SP (2014): http://www.etelg.com.br/robotica/torneio_robotica_2014.htm
10	ETEC Centro Paula Souza - Olímpia-SP - (2012): https://papobinario.wordpress.com/2012/04/27/campeonato-robocode-etec-de-olimpia/
11	ETEC Centro Paula Souza - Prof. José Ignácio Azevedo Filho - Ituverava-SP (2014): http://supervisaorpreto.blogspot.com.br/2014/10/alunos-do-etim-realizam-torceio-de.html
12	ETEC Centro Paula Souza - Professor Mário Antônio Verza - Palmital-SP (2014): http://www.etecpalmital.com.br/_noticias/_eventos/_eventos/_primeiroCampRobocode/primeiroCampRobocode.html
13	ETEC Centro Paula Souza - São Paulo-SP - (2014): http://www.robotica.cpscetec.com.br/informatica.php
14	FAAR - Faculdades Associadas de Ariquemes - Ariquemes-RO (2011): http://www.capitaldojerico.com.br/noticias/tecnologia/3090/faar-realiza-1-torneio-de-programacao-robocode.html
15	Faculdade Joaquim Nabuco - Paulista-PE (2012): http://www.joaquimnabuco.edu.br/noticia/exibir/cid/10/nid/1351/fid/1

16	FAGOC - Faculdade Governador Ozanam Coelho - Ubá-MG (2008): http://fagoc.br/noticias/curso-de-ciencia-da-computacao-da-fagoc-realiza-torneio-de-robos-entre-alunos
17	FT-UNICAMP - Faculdade de Tecnologia - Limeira-SP (2015, 2014, 2013, 2012, 2011): http://torneiorobocode.orgfree.com/torneio-ft.php
18	IBTA - Instituto Brasileiro de Tecnologia Avançada - Unidade Paulista - São Paulo-SP (2014): http://www.ibta.edu.br/db/Regulamento-Robocode.pdf
19	IC-UNICAMP - Instituto de Computação - Campinas-SP (2012): http://www.ft.unicamp.br/liag/wp/blog/sem-categoria/torneio-robocode-do-ic/
20	IESP - Instituto de Ensino Superior da Paraíba - Cabedelo-PB (2015): http://www.iesp.edu.br/sesp/?author=1&paged=46
21	IF Farroupilha - São Borja-RS (2015, 2014): http://www.sb.iffarroupilha.edu.br/site/conteudo.php?cat=13&sub=1855
22	IFAL - Instituto Federal de Alagoas - Arapiraca-AL (2015, 2014): http://www2.ifal.edu.br/noticias/ii-torneio-robocode-do-ifal-e-realizado-no-caiite-2015
23	IFBA - Instituto Federal Bahia - Campus Vitória da Conquista-BA (2013): http://www.conquista.ifba.edu.br/index.php/85-ifba/896-secitec-2013-comeca-na-proxima-semana
24	IFC - Instituto Federal Catarinense - Camburiú-SC (2012): http://linhapopular.com.br/novo/2013/10/22/ifc-realiza-encontro-de-tecnologia-e-informacao/
25	IFC - Instituto Federal Sul-Rio-Grandense - Bajé-RS (2012): http://www.bage.ifsul.edu.br/portal/index.php?option=com_content&view=article&id=182:atividades-no-campus-bage
26	IFES - Instituto Federal Espírito Santo - Colatina-ES (2012): http://www.colatina.es.gov.br/eventos/programacao_semana_tec_2009.pdf
27	IFG - Instituto Federal Goiás - Campus Luziânia-GO (2014): http://robocode.weebly.com/
28	IFPI - Instituto Federal Piauí - Campus Floriano-PI (2014): http://www.floriano.ifpi.edu.br:8099/site/modules/noticias/Noticia.mtw?id=415
29	IFPR - Instituto Federal Paraná - Umuarama-PR (2011): http://umuarama.ifpr.edu.br/wp-content/uploads/2015/04/3-CERTIFICADO-Ganhadores-do-desafios-de-rob%C3%B4s.pdf
30	IFRN - Instituto Federal do Rio Grande do Norte - Nova Cruz-RN (2013): http://www2.ifrn.edu.br/expotecnc/robocode/
31	IFSC - Instituto Federal Santa Catarina São José-SC - (2014): http://wiki.sj.cefetsc.edu.br/wiki/index.php?title=Torneio_RoboCode&oldid=37085
32	IFSP - Instituto Federal São Paulo - Boituva-SP (2015): http://btv.ifsp.edu.br/site/index.php/2-uncategorised/339-campus-boituva-promove-minicurso-e-torneio-sobre-robocode
33	IFSP - Instituto Federal São Paulo - Campus do Jordão-SP (2014) : http://snct.ifspcjo.edu.br/2014/index.php/evento/detalhepalestra/29
34	IFSP - Instituto Federal São Paulo - Capivari-SP (2015, 2014, 2011): http://www.ifspcapivari.com.br/2015/08/torneio-robocode/
35	PUC Minas - Pontifícia Universidade Católica - Guanhães-MG (2008): http://www.inf.pucminas.br/siguanhaes/

36	UDESC - Universidade do Estado de Santa Catarina - Ibirama-SC (2011): http://www2.joinville.udesc.br/~larva/sicgrapi/tutorialrobocode.pdf
37	UDESC - Universidade do Estado de Santa Catarina - Joinville-SC (2013): http://88.198.249.35/d/Robocode-Wiki.pdf
38	UFC - Universidade Federal do Ceará - Campus Pici - Fortaleza-CE (2007): http://www.linuxnarede.com.br/index2.php?subaction=showfull&id=1175953990
39	UFC - Universidade Federal do Ceará - Quixadá-CE (2011, 2007): https://robocodeufc.wordpress.com/evento/
40	UFMA - Univerisade Federal do Maranhão - São Luís-MA (2014, 2009): http://pet.ufma.br/computacao/?page_id=356
41	UFU - Universidade Federal de Uberlândia - Patos de Minas-MG (2013, 2012): https://www.facebook.com/permalink.php?id=380880845332260&story_fbid=382933938460284
42	UFU - Universidade Federal de Uberlândia - Uberlândia-MG (2013): http://www.ft.unicamp.br/liag/wp/blog/sem-categoria/ex-integrante-do-liag-em-congresso-em-minas/
43	UNIFAL - Universidade Federal de Alfenas - Alfenas-MG (2013): http://bcc.unifal-mg.edu.br/semanaacad2013/campeonato.php
44	UNIPAC - Faculdade Presidente Antônio Carlos - Contagem-MG (2011): http://www.folhadecontagem.com.br/portal/index.php/edicoes-da-semana-2011/275-edicao-672-11-a-17112011/4953-unipac-contagem-realiza-o-1o-campeonato-robocode.html
45	UNIPAR - Universidade Paranaense - Francisco Beltrão-PR (2014): http://presencial.unipar.br/noticias/2014/11/25/francisco-beltrao-unipar-promove-segundo-torneio-virtual-de-robos-virtuais/
46	UNISAL Lorena-SP – 2012: http://unisal.br/robocode-e-batalha-de-robos-desafiam-estudantes/
47	URI - Universidade Regional Integrada do Alto Uruguai e das Missões - Campus Santo Ângelo-RS (2012): http://www.ijui.com/educacao/39870-simposio-de-inovacao-em-tecnologias-computacionais-inicia-na-proxima-segunda.html
48	UTFPR - Universidade Tecnológica Federal do Paraná - Pato Branco - PR (2013): http://www.utfpr.edu.br/patobranco/estrutura-universitaria/assessorias/ascom/arquivos/documentos-2013/relacao-dos-projetos-expostos-na-expout-inventum-2013/Foto%201.jpg/view