UNIVERSIDADE FEDERAL DO AMAZONAS
FACULDADE DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

FDRobô: UM FRAMEWORK DIDÁTICO PARA AUXILIAR O ENSINO DE LINGUAGEM DE PROGRAMAÇÃO ADAPTADA AO MÉTODO DE APRENDIZAGEM COOPERATIVA E COMPETITIVA

WOLLACE DE SOUZA PICANÇO

MANAUS-AM
2016

UNIVERSIDADE FEDERAL DO AMAZONAS
FACULDADE DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

WOLLACE DE SOUZA PICANÇO

FDRobô: UM FRAMEWORK DIDÁTICO PARA AUXILIAR O ENSINO DE LINGUAGEM DE PROGRAMAÇÃO ADAPTADA AO MÉTODO DE APRENDIZAGEM COOPERATIVA E COMPETITIVA

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Amazonas, como requisito parcial para obtenção do título de Mestre em Engenharia Elétrica, área de concentração: Automação de Sistemas.

Orientador:

Prof. Dr. –Ing. Vicente Ferreira de Lucena Júnior

MANAUS
2016

WOLLLACE DE SOUZA PICANÇO

FDRobô: UM FRAMEWORK DIDÁTICO PARA AUXILIAR O ENSINO DE LINGUAGEM DE PROGRAMAÇÃO ADAPTADA AO MÉTODO DE APRENDIZAGEM COOPERATIVA E COMPETITIVA

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Amazonas, como requisito parcial para obtenção do título de Mestre em Engenharia Elétrica, área de concentração Automação de Sistemas.

Aprovado em 29 de janeiro de 2016.

BANCA EXAMINADORA

Prof. Dr. –Ing. Vicente Ferreira de Lucena Junior
Universidade Federal do Amazonas – UFAM

Prof. Dr. José Francisco de Magalhães Neto
Universidade Federal do Amazonas – UFAM

Prof. Dr. Antonio da Fonseca de Lira
Instituto Federal de Educação, Ciência e Tecnologia do Amazonas - IFAM

Ficha Catalográfica

P585f

Picanço, Wollace de Souza

FDRobô: Um *Framework* Didático Para Auxiliar O Ensino De Linguagem De Programação Adaptada Ao Método De Aprendizagem Cooperativa E Competitiva / Wollace de Souza Picanço. 2016 105 f.: 31 cm.

Dissertação (Mestrado em Engenharia Elétrica, área de concentração em Automação de Sistemas) — Universidade Federal do Amazonas, 2016.

Orientador: Prof. Dr. Ing. Vicente Ferreira de Lucena Junior

1. *Framework* 2. Aprendizagem Cooperativa. 3. Aprendizagem Competitiva. 4. Education. I. Lucena Junior, Vicente Ferreira de (Orient.) II.Universidade Federal do Amazonas III. Título

Dedico este trabalho à minha mãe Arlete de Souza Picanço, aos meus amigos do grupo de TV Digital da UFAM e a todos aqueles que lutam pela melhoria do processo de ensino-aprendizagem.

Agradecimentos

Este trabalho é de grande importância para minha carreira, pois representa o fim de um ciclo de muito aprendizado determinando a próxima direção da minha vida. Portanto quero agradecer de coração primeiramente a Deus pela sabedoria e perseverança para superar os momentos de dificuldades que passei e a todos aqueles que compartilharam comigo em especial:

A meus amigos do grupo de TV Digital Hiran Amaral, Ricardo Rosa, Vandermi Silva e Walter Simões.

A meus amigos do Centro Universitário do Norte Wanderlan Albuquerque, Anderson Esteves e Ricardo Barbosa.

A todos os professores das disciplinas que cursei.

Ao professor orientador Vicente por todo apoio e incentivo.

A toda minha família: e em especial a minha mãe Arlete de Souza Picanço, minhas irmãs Jane de Souza Moreira, Bia Coelho e, meu sobrinho Leandro Picanço.

À minha filha Taliany da Silva Picanço pela fonte de inspiração e motivação.

“Às vezes a vida te bate com um tijolo na cabeça. Não perca a fé. Estou convencido de que a única coisa que me fez continuar foi que eu amava o que eu fazia. Você precisa encontrar o que você ama. E isso vale para o seu trabalho e para seus amores. Seu trabalho irá tomar uma grande parte da sua vida e o único meio de ficar satisfeito é fazer o que você acredita ser um grande trabalho. E o único meio de se fazer um grande trabalho é amando o que você faz”.

Steve Jobs

Resumo

A produção de novas abordagens utilizando os recursos tecnológicos no processo de ensino-aprendizagem vem crescendo no ambiente educativo. Por conseguinte, o uso de aplicações educativas voltada para a robótica, vem facilitando a aprendizagem na educação. No entanto, alunos de engenharia, especificamente na disciplina de linguagens de programação encontram dificuldades em aprender o método padronizado que comunicar instruções para um computador. Esta dissertação aborda o uso de *frameworks* dedicados e métodos de aprendizagem cooperativa-competitiva com a finalidade de facilitar o processo de ensino de linguagens de programação. Além disso, visa definir uma arquitetura de *software* para manipular um robô no ambiente real de aprendizagem. O propósito é promover uma ferramenta que auxilie na interpretação das instruções de linguagem de programação por meio de um robô. O processo de definição da arquitetura e do conjunto de métodos para manipular o robô, foi realizado para auxiliar as dificuldades encontradas nos estudantes da disciplina de linguagens de programação. A partir dessas verificações, foi possível elaborar um modelo de sistema, específico para linguagem de programação C. O protótipo construído neste trabalho se diferencia de produtos comerciais, pois se levou em consideração a adaptação do método de aprendizagem cooperativa e competitiva. Buscou-se oferecer uma opção que facilitasse a edição do código-fonte e de sua interpretação quanto ao seu uso no processo de ensino-aprendizagem. O processo de avaliação do protótipo foi realizado por meio dos métodos da Engenharia da Usabilidade. Os resultados obtidos nos experimentos demonstram que o ambiente contribuiu para o processo de ensino-aprendizagem de linguagens de programação. Portanto, os critérios de medição da usabilidade estabelecidos pela norma ISO 9241, afirmam que a ferramenta proposta é mais um apoio de recursos tecnológicos que pode ser utilizada na educação.

Palavras-chave: *Framework*, Aprendizagem Cooperativa, Aprendizagem Competitiva, Educação.

Abstract

The production of new approaches using the technological resources in the teaching-learning process has grown in the educational environment. Therefore, the use of educational applications focused on robotics, is facilitating learning in education. However, engineering students, specifically in the discipline of programming languages are difficult to learn the standardized method to communicate instructions to a computer station. This dissertation addresses the frames of dedicated resources and methods of cooperative-competitive learning in order to facilitate the process of learning programming languages. It also seeks to define a software architecture to manipulate a robot in the real learning environment. The purpose is to promote a tool to assist in interpreting programming language instructions by a robot. The architecture definition process and set of methods to manipulate the robot was held to assist the difficulties encountered in students the discipline of programming languages. From these findings, it was possible to develop a system model, specific to the C programming language The prototype built in this paper is different from commercial products, because it took into account the adaptation of cooperative and competitive learning method. We attempted to offer an option that facilitates the editing of source code and its interpretation as to its use in the teaching-learning process. The evaluation process of the prototype was carried out by the methods of Usability Engineering. Based on results of experiments show that the environment has contributed to the process of teaching-learning programming languages. Therefore, the usability of the measurement criteria established by the ISO 9241 standard, claim that the proposal is another tool I support technological resources that can be used in education..

Keywords: Framework; Collaborative Learning; Competitive Learning, Education.

Índice de Figuras

Índice de Quadros

Lista de Siglas

API	- *Application Programming Interfaces*
AP	- *Access Point*
GIF	- *Graphics Interchange Format*
HTML	- *HyperText Markup Language*
HTTP	- *Hyper Text Transfer Protocol*
HRI	- Interação Robô Humano
IPTV	- *Internet Protocol-Television*
IHC	- *Interface Humano-Computador*
ISO	- *International Organization for Standardization*
JCIC	- *Joint Committee on InterSociety Coordination*
JVM	- *Java Virtual Machine*
JMF	- *Java Media Framework*
JDK	- *Kit de Desenvolvimento Java*
LP	- *Linguagem de Programação*
OO	- *Programação Orientada a Objeto*
PC	- *Paralisia Cerebral*
TCP	- *Transmission Control Protocol*
RTC	- *Robot Task Commander*
TEA	- *Transtorno Espectro Autismo*
UP	- *Processo Unificado*
UML	- *Unified Modeling Language*
USA	- *United States of America*
MPEG	- *Moving Pictures Expert Group*
XML	- *Extensible Markup Language*
WBT	- *Web Based Training*

Glossário

Linguagem de Programação	LP - Abreviatura designada para uma linguagem de programação, ou seja, um paradigma de programação fornece e determina a visão que o programador possui sobre a estruturação e execução do programa. Por exemplo, em programação orientada a objetos, os programadores podem abstrair um programa como uma coleção de objetos que interagem entre si.
Orientação Objeto	OO - Sigla utilizada para abreviar a programação Orientada a Objeto é um modelo de análise, projeto e programação de sistemas de *software* baseado na composição e interação entre diversas unidades de *software* chamadas de objetos.
Processo Unificado	UP - Sigla que representa um modelo de Processo Unificado na engenharia de *software*. Surgiu para realizar o desenvolvimento de *software* visando à construção de sistemas orientados a objetos.
International Organization for Standardization	ISO - Utilizada para representar a Organização Internacional para Padronização. A ISO tem como objetivo principal aprovar normas internacionais em todos os campos técnicos, como normas técnicas, classificações de países, normas de procedimentos e processos, e etc.
United States of America	USA - Estado Unidos da America uma república constitucional federal composta por 50 estados e um distrito federal. A maior parte do país situa-se na região central da América do Norte, formada por 48 estados.
Robot Task Commander	RTC - projeto desenvolvido para um sistema de computador que utiliza um robô para resolução de tarefas.
Transtorno Espectro Autismo	TEA - Abreviatura para Transtorno Espectro Autismo tipo de síndrome que afeta o ser humano. Esta Sigla foi utilizada em um trabalho científico.
Interação Robô Humano	HRI - Abreviatura de Interação Robô Humano denominado em um projeto de engenharia e referenciado neste projeto de dissertação.

Paralisia Cerebral	PC - Abreviatura designada para Paralisia Cerebral é uma lesão de uma ou mais partes do cérebro, provocada muitas vezes pela falta de oxigenação das células cerebrais.
Interação Homem-Robô	PARO - Abreviatura para Interação Homem-Robô definido em um projeto de automação e referenciado neste projeto de dissertação.
Access Point	AP - Abreviatura de (Access Point) tipo de mecanismo de comunicação é um componente de uma rede sem fio que realiza a interconexão entre todos os dispositivos móveis.
Transmission Control Protocol	TCP - Abreviatura para o Protocolo de Controle de Transmissão de dados, é um dos principais protocolos da camada de transporte do modelo TCP/IP.
Graphics Interchange Format	GIF - Abreviatura de Graphics Interchange Format, extensão de um tipo de imagem ou é um formato de dados utilizado para imagens.
Unified Modeling Language	UML - Abreviatura para (Unified Modeling Language), Pela definição de seu nome, vemos que UML é uma linguagem que define uma série de artefatos que nos ajuda na tarefa de modelar e documentar os sistemas orientados a objetos que desenvolvemos.
Interface Humano-Computador	IHC - Abreviatura de Interface Humano-Computador, é uma matéria interdisciplinar que relaciona a ciência da computação, artes, design, ergonomia, psicologia, sociologia, semiótica, linguística, e áreas afins.
Kit de Desenvolvimento Java	JDK - Sigla definida para o Kit de Desenvolvimento Java, um conjunto de utilitários que permitem criar sistemas de software para plataforma Java.
Application Programming Interface	APIs - Abreviatura de Interface de Programação de Aplicações é um conjunto de rotinas e padrões de programação para acesso a um aplicativo de *software*.

Sumário

Capítulo 1- Introdução

O processo de ensino-aprendizagem, seja a distância ou presencial, alcançou mudanças significativas no novo cenário mundial graças ao avanço tecnológico. Embora, ainda haja carência de recursos pedagógicos que ajudem a promover a aprendizagem por meio de experimentos práticos, é notório o crescimento do uso de aplicativos na robótica educativa.

Para os pesquisadores da PUCRS a robótica na educação também conhecida como Robótica Pedagógica, é caracterizada por ambientes de aprendizagem onde o aluno pode construir os conceitos por meio da montagem e da programação de robôs [42]. Atualmente existem vários projetos pedagógicos utilizando robôs como, projeto Lego, projetos Vex e projeto Robotino da FESTO® Ltda [27, 10, 58].

Aliado a esses projetos foram incorporados novos aplicativos para o processo de ensino-aprendizagem de linguagens de programação. De acordo com Albuquerque, os grandes desafios enfrentados no ensino de linguagem de programação é a necessidade de uso de métodos de aprendizagem que permitam tornar o processo de ensino mais efetivo [3].

Esta dissertação trata das dificuldades dos estudantes de engenharia, no que se refere à compreensão das instruções apresentada pela linguagem de programação C. C é uma linguagem de programação padronizada pela ISO e existem poucas arquiteturas para as quais não existem compiladores para C [61]. Apesar da linguagem de programação C ter essas qualidades, existe uma questão relevante a considerar: Como utilizar um recurso tecnológico visando facilitar a compreensão das linguagens de programação?

Esta abordagem integra os conceitos de *framework*, robótica educativa e método de aprendizagem cooperativa e competitiva para facilitar a compreensão de aprendizagem de linguagem de programação. Além disso, visa produzir um *framework* denominado de "FDRobô" com o objetivo de auxiliar o estudante de engenharia a construir o conhecimento por meio de um robô móvel.

Dessa forma, o "FDRobô" será avaliado por meio dos seguintes passos. No primeiro passo aplicou-se um conjunto de atividades visando extrair o rendimento dos alunos. No segundo passo, fez-se o uso de dois princípios estabelecidos pela Norma ISO

9241- Parte 10 – Princípios de Diálogo para medir a satisfação geral e pedagógica do aluno em relação ao *framework* proposto. Assim, o primeiro princípio baseou-se no modelo de questionário proposto por Nielsen [35]. O segundo princípio foi o modelo de questionário proposto por Nokelainen [37].

Os resultados obtidos na experiência demonstraram que o ambiente contribuiu positivamente para o aproveitamento da turma de estudantes. Já para inferir a eficácia da interação humano-computador face a efetiva realização das tarefas e a satisfação ou insatisfação (efeito subjetivo) que ela possa trazer de forma didática obteve-se um resultado satisfatório tanto para a usabilidade geral quanto para usabilidade pedagógica. No restante deste capítulo são apresentadas a motivação, objetivos gerais e específicos, a metodologia usada e a organização do restante do trabalho.

1.1 Motivação/Justificativa

Linguagens de programação envolvem abstrações conceituais complicadas para a compreensão do aluno recém chegado a universidade. Por exemplo, é fácil criar uma variável na linguagem de programação C, observe:

```
void main ()
{       int a;
        a = 2;
        printf ("Valor da Variável: %d", a);  }
```

A variável inteira “a” recebeu o valor dois (2) e o resultado desse programa é um texto na tela do seu computador mostrando: “**Valor da Variável: 2**”. Mas, se esta variável fosse responsável pelo deslocamento de uma aeronave ou de um braço robótico, o programa teria um nível de instruções bastante complicado.

A ideia de desenvolver esse *framework* como ferramenta de apoio ao processo de ensino-aprendizagem, origina-se da falta de experimentos práticos em aulas de linguagem de programação. Para Savi, uma forma de minimizar essa situação é o uso de métodos de ensino alternativos, como, estudos de caso, atividades realizadas em projetos de empresas do ramo, jogos (cartas, tabuleiro, computador, etc.), simuladores, entre outros [48].

A inserção de recursos tecnológicos como forma de auxílio na educação é assunto de grande debate no Brasil. Conforme as pesquisas relatadas em RobEduc demonstram nos países de primeiro mundo que esse debate já foi superado, pois a maioria da população já

tem acesso a recursos como computador, internet e programas educativos na escola e até na própria residência [45].

Por outro lado, a realidade brasileira aponta para o uso intenso de soluções livres, abrindo assim um campo interessante para disseminação de recursos tecnológicos a baixo custo para governos e entidades educacionais. Atualmente, o computador é utilizado como ferramenta de captação de informações, ou seja, uma biblioteca mais fácil, rápida e atrativa que as bibliotecas tradicionais.

Assim, estas bibliotecas aliam os computadores a programas específicos para o processo de ensino-aprendizagem, por exemplo, a robótica equipa os laboratórios com estruturas que podem auxiliar a aprendizagem com eficiência e eficácia. Para pesquisadores de RobEduc, a robótica educacional procura auxiliar o aluno na construção do aprendizado adquirido em sala de aula [45], já que é nela que são transformadas em idéias que estimulam o aluno a sempre querer aprender mais e instiga a vontade de absorver novos conhecimentos.

O *framework* proposto nesta dissertação utiliza um robô contendo métodos reservados para construir o conhecimento da disciplina de linguagem de programação na prática. Assim, um novo exemplo da instruçã em C executado anteriormente, ficaria da seguinte forma:

```
void main ()
{       int moveFrente;
        moveFrente = 2;
        printf (moveFrente); }
```

Observe agora, que a variável **moveFrente** continua sendo inteira, porém o **moveFrente** é uma palavra reservada do robô. O resultado desse programa é a movimentação do robô em ambiente real. Além disso, os alunos são submetidos ao método de aprendizagem cooperativa e competitiva visando prepará-los ao mundo do trabalho.

Motivados por essa ideia, tem-se a possibilidade de proporcionar ao cérebro do aluno, itens como, perceber, aprender, recordar e pensar sobre toda informação captada por meio dos sentidos. Assim, verifica-se a oportunidade de investigar futuras arquiteturas de programas de computador para fins de pesquisas na área da educação.

1.2 Problema

As linguagens de programação envolvem conceitos complicados para a compreensão do aluno recém chegado à universidade. Problemas como o não entendimento da sintaxe e de conceitos fundamentais de linguagens de programação, resultam na desmotivação dos alunos iniciantes e distanciando os alunos menos aptos desta área de conhecimento de tal modo que aumenta a taxa de desistência.

Esse problema pode ser tratado por meio de experimentos práticos em laboratórios destinados às eliminações de dúvidas e de construção de conhecimento [48]. Para isso, a robótica educacional proporciona ao educando o estudo de conceitos multidisciplinares, como física, matemática, engenharia, entre outros, existindo variações no modo de aplicação por meio da interdisciplinaridade [45].

Em virtude disso, a questão de investigação do presente trabalho é: Como o *framework* (FDRobô) pode auxiliar o aluno na aprendizagem dos conceitos de Linguagem de Programação? A hipótese proposta é: O desenvolvimento de um *framework* para o ensino de linguagem de programação C e a adaptação ao método da Aprendizagem Cooperativa e Competitiva poderia minimizar esse problema.

1.3 Objetivo Geral

O principal objetivo desta pesquisa é propor um *framework* para auxiliar a aprendizagem de linguagem de programação C de alunos de engenharia e que utilizará um robô móvel como instrumento prático de fixação de conceitos por meio do método de aprendizagem cooperativa e competitiva.

1.4 Objetivos Específicos

- Investigar os *frameworks* para o processo de ensino existente, identificando princípios, regras, etapas, atividades, ferramentas e artefatos;
- Propor um *framework* auxiliado por meio de um robô;
- Adaptar o *framework* ao método de aprendizagem cooperativo e competitivo
- Elaborar um estudo de caso para testar o *framework* com os alunos;
- Avaliar os resultados.

1.5 Metodologia

A metodologia utilizada para o desenvolvimento deste trabalho de pesquisa foi o Vê Epistemológico de Gowin [13]. A Figura 1.1 descreve que cada uma das dimensões do Vê evidencia a ocorrência de uma interação entre os domínios conceitual (Pensar) e a metodologia (Fazer).

Assim, faz-se necessário descrever as questões consideradas básicas as quais ajudam o desenvolvimento desse trabalho, do lado esquerdo da dimensão do Vê tem-se o "Pensar", onde se descreve as questões em foco, que virão explicitar claramente o que o trabalho se propõe realizar.

As Linguagens de Programação envolvem conceitos complicados para a compreensão do aluno recém chegado a universidade. Como o *framework* pode auxiliar o aluno a construir o conhecimento relacionado à Linguagem de Programação?

A questão em foco desenvolve "um *framework* e adapta ao método de aprendizagem cooperativo e competitivo por meio de um robô. A garantia de que o *framework* vai proporcionar esses itens está na fundamentação teórica descrita nos princípios do Vê.

Essas descrições referem-se aos métodos que são eficazes para a obtenção e a interpretação do *framework* proposto. Define-se também os conceitos-chave, visto que esses conceitos são fundamentais para a compreensão da dissertação. Já o evento descreve os experimentos utilizados para o desenvolvimento da estrutura do *framework.*

Já do lado direito da dimensão do Vê tem-se o "Fazer", onde se obtém os dados (registro) da pesquisa visando analisá-los (Transformações). Esse analise acontecerá de quadro contendo o tempo das tarefas de cada equipe. O resultado da avaliação do protótipo é descrita na satisfação produzida e nas asserções de valor. Elas afirmam, esclarecem e concluem sobre a qualidade ou valor do questionamento abordado.

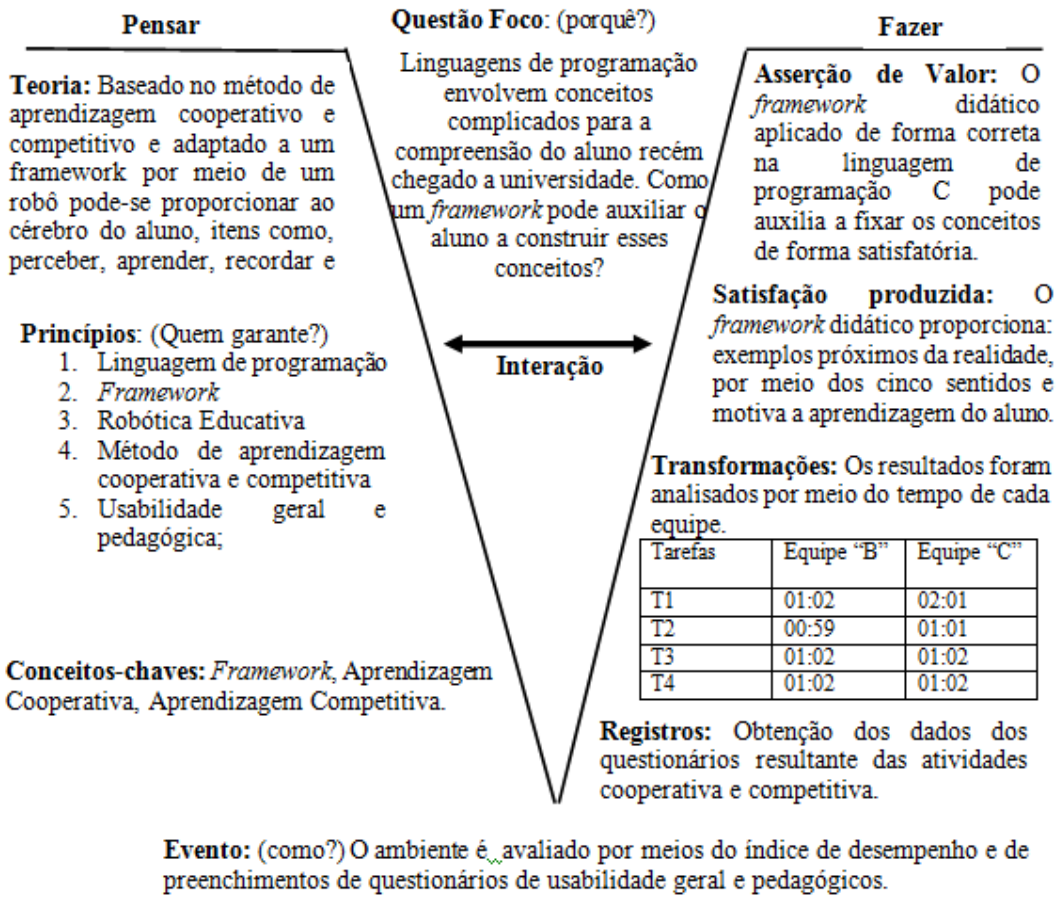


Figura 1.1 - Metodologia do Trabalho (Vê Epistemológico – Gowin [13])

1.6 Organização do Trabalho

Esta dissertação de mestrado está estruturada em sete capítulos, sendo esta a introdução. Nela apresentou-se um retrato inicial (motivação, justificativa, problema e metodologia) do que será desenvolvido ao longo de todo o estudo.

O capítulo 2 destina-se à fundamentação teórica, onde são abordados conceitos relevantes para esta dissertação como, nível e paradigma de linguagens de programação, conceitos, característica e classificação de *framework*, robótica educativa, método da aprendizagem cooperativa/competitiva e usabilidade geral e pedagógica.

No capítulo 3 são explanados trabalhos considerados relevantes no processo de definição das técnicas e da construção dos protótipos desta pesquisa, procurando ressaltar características de estruturas de *frameworks*, robótica e o método de aprendizagem cooperativa e competitiva.

O capítulo 4 apresenta a concepção da solução desejada. Assim um conjunto de atividades será desenvolvido neste capítulo como, especificação de requisitos, análise de componentes, adaptações de requisitos, projeto de arquitetura do *framework* para o desenvolvimento do protótipo.

O capítulo 5 mostra a implementação do modelo proposto. Neste capítulo é desenvolvido um protótipo, desenvolvimento de um conjunto de métodos para manipular o robô. Foram também definidos equipamentos e *softwares* que garantam que os objetivos específicos sejam alcançados, seguindo o modelo de concepção do Capítulo 4.

O capítulo 6 expõe o processo de avaliação do *Framework*. Nesta etapa foram selecionados estudantes para avaliar e testar o protótipo por meio de questionários de usabilidade geral e pedagógico.

O capítulo 7 é destinado às considerações finais como, dificuldades encontradas e trabalhos futuros. Finalmente o presente trabalho é encerrado pelas referências bibliográficas**,** publicação, apêndices e anexos.

Capítulo 2- Fundamentação Teórica

A fim de compreender os conceitos e soluções discutidas neste trabalho, é necessário compreender como funcionam algumas técnicas, tecnologias e como estas se ajustam e se relacionam para serem utilizadas em conjunto para a construção do *framework* proposto.

O capítulo inicia com uma explanação sobre os conceitos básicos de linguagens de programação no contexto de classificação (nível e paradigma), sintaxe e semântica. Em seguida, são apresentadas as abordagens que apóiam o reúso de *software*, no entanto, o foco é específico no estudo de *frameworks* que influenciam o processo de desenvolvimento de arquiteturas reusáveis.

Após isso, se exploram os conceitos de robótica educativa destacando a importância de alguns projetos desenvolvidos para o ensino da robótica. Em seguida, apresentam-se os conceitos dos métodos de aprendizagens cooperativas e competitivas e como estes podem ser utilizados no processo de ensino-aprendizagem. Por fim são selecionados e apresentados alguns métodos de Usabilidade visando utilizá-los na validação desse projeto.

2.1 Linguagem de Programação

Antes do surgimento das Linguagens de Programação (LP), a programação de computadores era feita exclusivamente em linguagem de máquina, ou seja, os programadores tinham de conhecer profundamente a arquitetura da máquina na qual o programa seria executado, seu conjunto de instruções e sua forma de funcionamento. Sebesta afirma que mesmo dominando todo esse conhecimento, a atividade de programação era pouco produtiva porque as instruções das linguagens de máquinas eram muito simples [49].

Para Tucker e Noonan as primeiras LP (FORTRAN e COBOL) surgiram no final dos anos 50 e início dos anos 60 visando facilitar o trabalho de programação [56]. À medida que os recursos computacionais eram desenvolvidos, os computadores iam se tornando mais poderosos e úteis. Assim, a atividade de programação se tornava um gargalo

para a disseminação dos sistemas computacionais. E no final dos anos 60 surgiram as LP que enfatizavam a programação estruturada (PASCAL e C são exemplos) [56].

Autores de AdaCore, relatam o aumento da complexidade dos sistemas computacionais, a abstração de dados passou a ser o foco das linguagens de programação e no início dos anos 80 surgiram a construção modularizada de programas e bibliotecas (MODULA-2 e ADA são exemplos) [2].

Nos anos 80 e 90 a disseminação do uso dos computadores pessoais e das estações de trabalho possibilitou o surgimento da indústria de *software*, e com ela, a necessidade de se classificar, produzir e atualizar *software* rapidamente. Para esse fim, surgem também as metodologias orientadas a objetos (Smalltalk, C++ e JAVA, são exemplos) [52].

2.1.1 O Ensino de Linguagens de Programação

O ensino de linguagens de programação, nas escolas e faculdades voltadas para a formação de profissionais da área de informática, vem obedecendo à mesma linha metodológica tradicional. Em geral, os livros técnicos e apostilas contêm modelos prontos de programas muito elementares, aos quais se seguem algumas séries de exercícios, que não passam de variações dos modelos.

Um exemplo muito comum é o *"Hello World"*, que habitualmente aparece nas literaturas como um dos primeiros programas a ser desenvolvido. Os alunos são exaustivamente orientados a elaborar esse programa em diversas linguagens, mas isso não significa que eles estejam entendendo a função do programa, que é escrever qualquer mensagem ou frase na tela.

Autores como Fontes e Silva afirmam que essa metodologia de ensino talvez fosse satisfatória quando os programas de computadores apresentavam uma *interface* com o usuário, por meio de telas com informações apenas sob a forma de textos, e os objetivos em termos de programação fossem muito limitados [9].

O aluno que estiver estudando sob essa metodologia não terá oportunidades de aprendizagem para desenvolver sua capacidade de compreensão e abstração do pensamento lógico-computacional, pois a literatura técnica disponível emprega exemplos prontos, distantes de sua realidade. Ao ingressar no mercado de trabalho, em geral como estagiário de programação, esse aluno encontrará dificuldade em relacionar o conhecimento recebido com as necessidades reais da empresa.

2.1.2 Classificação das Linguagens de Programação

Ao longo dos anos, uma grande quantidade de linguagens de programação foi desenvolvida (e continua sendo), algumas de uso mais geral e outras concebidas para áreas de aplicação específicas. Para Sebesta [49], as linguagens de programação são classificadas tanto em níveis de linguagens (sendo que as linguagens de nível mais baixo são mais próximas da linguagem interpretada pelo processador e mais distantes das linguagens naturais), quanto em nível de paradigma. O mesmo autor define o nível de linguagens da seguinte forma:

Linguagem de Programação de Baixo Nível: são aquelas voltadas para a máquina, ou seja, as que são escritas utilizando as instruções do microprocessador do computador e são genericamente chamadas de linguagens Assembly.

Linguagem de Programação de Médio Nível: pode-se acessar aos registros do sistema e trabalhar com endereços de memória. Vale ressaltar que estas linguagens de programação realizam operações tanto de baixo nível, quanto de alto nível (*if...else; while; for*). Por exemplo, Linguagem C.

E finalmente, Linguagem de Programação de Alto Nível: são linguagens voltadas para o ser humano. Em geral utilizam sintaxe mais estruturada tornando o seu código mais fácil de entender e de editar programas. Estas linguagens permitem ao programador se esquecer completamente do funcionamento interno da máquina para a qual ele está desenvolvendo o programa. Somente necessitam de um tradutor que entenda o código fonte para as características da máquina. As linguagens C++, .NET e o JAVA são exemplos de linguagens de alto nível.

Algumas linguagens foram criadas durante a história da computação e outras foram adaptadas às novas formas de se pensar sobre programação, resultando em formas distintas de modelagem de soluções para problemas de *software*. Dessa forma, Sebesta define o paradigma como um modelo interpretativo (ou conceitualização) de uma realidade [49], ou seja, pode dizer-se que um paradigma é um ponto de vista que determina como uma realidade é entendida e como se atua sobre ela. Assim, temos as seguintes formas de paradigmas segundo este autor:

Imperativo: As linguagens imperativas (Figura 2.1) são orientadas a ações, onde a computação é vista como uma sequência de instruções que manipulam valores de variáveis (leitura e atribuição). Em outras palavras, paradigma também denominado de procedural,

por incluir subrotinas ou procedimentos como mecanismo de estruturação. Por exemplo, linguagens C, Python, Pascal e etc.

Figura 2.1 - Modelo computacional Imperativo

Funcional: Trata a programação como uma transformação de dados por funções, ou seja, que função deve ser aplicada para transformar minha entrada na saída desejada? Por exemplo, ML (Linguagem funcional fortemente tipada), Miranda (baseada em ML) e etc. A Figura 2.2, mostra o modelo funcional em diagrama de bloco.

Figura 2.2 - Paradigma do modelo Funcional

Lógico: O modelo Lógico está relacionado à perspectiva da pessoa, ou seja, ele encara o problema de uma perspectiva lógica. Um programa lógico é equivalente à descrição do problema expresso de maneira formal, similar à maneira que o ser humano raciocinaria sobre ele. Programação é baseada em fatos, que podem ser relações (associações) entre coisas, e regras, que produzem fatos deduzidos a partir de outros. Observe a relação (Maria é progenitora de Júlia sua antepassada) dos blocos no modelo Lógico, Figura 2.3. A linguagem Prolog é um exemplo desse paradigma.

Figura 2.3 - Paradigma do modelo Lógico

Orientado a Objetos: Tratam os elementos e conceitos associados ao problema como objetos. Objetos são entidades abstratas que embutem dentro de suas fronteiras, as características e operações relacionadas com a entidade real. O modelo Orientado a Objeto (OO) focaliza mais o problema. Um programa OO é equivalente a objetos que trocam mensagens entre si. Os objetos do programa equivalem aos objetos da vida real (problema). O paradigma OO considera objetos e classes como blocos básicos de

construção de um sistema. Sistemas são vistos como coleções de objetos que se comunicam, enviando mensagens, colaborando para representar o comportamento global dos sistemas. A Figura 2.4 mostra que o estado/operação da saída vai depender da entrada. Exemplo desse paradigma: Smalltalk, C++, Java e etc.

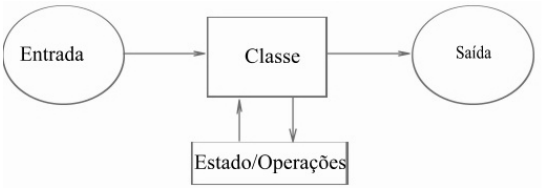


Figura 2.4 - Modelo do paradigma orientado a objetos

Concorrente: é um paradigma de programação para a construção de programas de computador que fazem uso da execução concorrente (simultânea) de várias tarefas computacionais interativas, que podem ser implementadas como programas separados. Essas tarefas podem ser executadas por um único processador, vários processadores em um único equipamento ou processadores distribuídos por uma rede. Para Hermann a programação concorrente é relacionada com programação paralela, mas foca mais na interação entre as tarefas. A interação e a comunicação correta entre as diferentes tarefas, além da coordenação do acesso concorrente aos recursos computacionais são as principais questões discutidas durante o desenvolvimento de sistemas concorrentes [15]. A Figura 2.5 mostra a solicitação de uma tarefa e em seguida o processamento concorrente implementa sua invocação. Exemplo de linguagens concorrentes Java e C#.

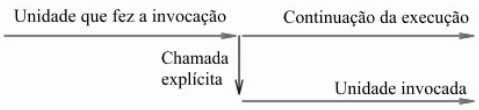


Figura 2.5 - Processamento concorrente

Hoje, existem diversos tipos de linguagens de programação e todas com suas características específicas. Para Sebesta, uma linguagem de programação é uma linguagem destinada a ser usada por uma pessoa para expressar um processo por meio do qual um computador pode resolver um problema [49]**.** Assim, a eficiência na construção e execução de programas depende da combinação desses quatro pontos de vista (pessoa, processo, computador e problema) que define como as instruções podem ser associadas visando à resolução de um determinado problema.

Assim, descreveram-se os elementos fundamentais de linguagens de programação para esta pesquisa. Na seção a seguir serão detalhadas as tecnologias de reutilização que mais se relacionam com a abordagem apresentada nesta dissertação de mestrado.

2.2 Framework

Para Sommerville o reúso de *software* é uma estratégia da engenharia, em que o processo de desenvolvimento é orientado para o reúso de *software* existente [53]. A Quadro 2.1 descrito pelo mesmo autor, demonstra algumas, das possíveis maneiras de implementação de reúso de *software:*

Quadro 2.1 - Abordagens que apóiam o reúso de *software*. Fonte: Sommerville [53].

Abordagem	**Descrição**
Padrões de Arquitetura	Padrões de Arquitetura de software que oferecem suporte a tipos comuns de sistemas de aplicação são usados como base de aplicação. Capítulos 6,7 e 20.
Padrões de Projetos	Abstrações genéricas que ocorrem em todas as aplicações são representadas como padrões de projetos, mostrando os objetos abstratos e concretos e as interações. Capítulos 7.
Desenvolvimento Baseado em componentes	Sistemas desenvolvidos através da integração de componentes (coleção de objetos) que atendem aos padrões de modelos e componentes. Capítulos 17.
Framework de Aplicação	Coleções de classe abstrata e concreta são adaptadas e estendidas para criar sistemas de aplicação. Subseção 2.2.1 desta pesquisa.

Para Pressman um *framework* é uma estrutura genérica estendida para se criar uma aplicação ou subsistema mais específico como um conjunto integrado de artefatos de *softwares* definidos em classes, objetos e componentes que colaboram para fornecer uma arquitetura reusável para uma família de aplicações relacionadas [41].

Para Sommerville “um *framework* é um conjunto de objetos que colaboram entre si, com o objetivo de atender um conjunto de responsabilidades para uma aplicação específica em um dado domínio” [53]. Albuquerque afirma que um *framework* é utilizado por meio de conexão de classes concretas e derivação de novas classes a partir de classes abstratas que podem fazer partes do próprio *framework* e/ou que estejam sendo utilizadas para o suporte de uma aplicação [3].

Assim, pode-se afirmar que um *framework*, é um modelo semi-acabado que pode originar outros tipos de artefato de *software*, além de aplicações, como outros *frameworks*, por exemplo. Ao longo do presente trabalho, quando se afirma que um *framework* é voltado à produção de aplicações, está-se considerando que mesmo utilizando um *framework* para desenvolver outros tipos de artefatos (é qualquer item criado como parte

da definição, manutenção ou utilização de um processo de software), o final da cadeia de desenvolvimento sempre resulta em aplicações.

Para Sommerville existem alguns benefícios na utilização de uma estrutura no formato de *framework* a qual o propósito final de um *framework* é ser reusável, mas para isso deve ser primeiro usável e bem documentado [53]. Assim, é necessário seguir alguns requisitos necessários, estabelecidos nas características básicas, segundo esse autor:

O *framework* deve ter funcionalidade abstrata (sem implementação) que deve ser completada e o desenvolvedor de aplicações não pode destruir o *framework.* A reusabilidade dos requisitos existentes no *framework* deve possibilitar os conceitos de extensibilidade. Já modularidade vai permite ao desenvolvedor elaborar aplicações que sejam distribuídas em formato de pacotes de maneira que organize diversas aplicações.

Um dos grandes benefícios a serem adquiridos pelo emprego do *framework* é a facilidade de tratamentos de erro. O *framework* é desenvolvido para tratar o problema proposto em um campo específico, ou seja, um domínio específico. A ideia básica para construção de um *framework* é não desenvolver uma solução para uma aplicação específica, mas sim capturar o comportamento geral de um domínio de aplicação e montar uma estrutura de controle capaz de representá-lo.

A realização de um *software* para uma implementação específica consiste em instanciar o referido *framework*, por meio da especialização de seus componentes. Esta especialização pode ser entendida como customização e extensão da estrutura do *framework*. A Figura 2.6 compara o processo de desenvolvimento de *software* utilizando *frameworks*.

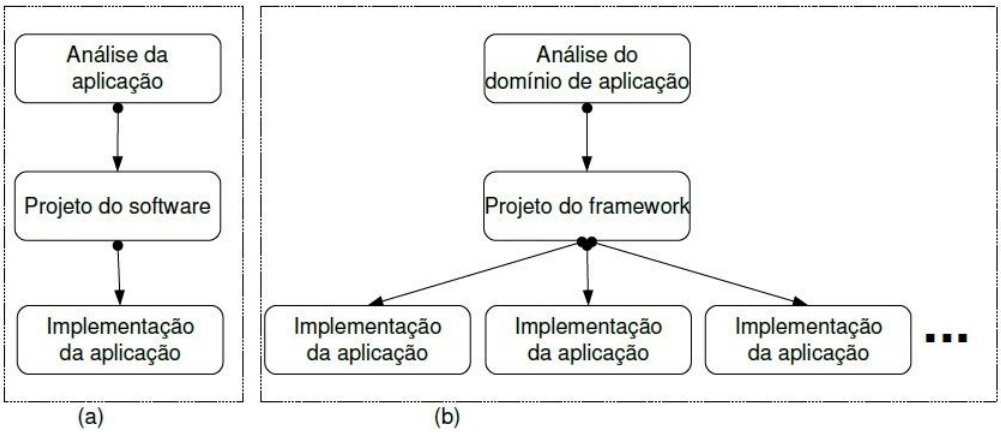


Figura 2.6 - Comparação dos processos de desenvolvimento utilizando puramente o paradigma de orientação a objetos (a) e *frameworks* (b). Fonte: Albuquerque 2012.

2.2.1 Classificação de Frameworks

Segundo Albuquerque a classificação de *frameworks* pode se dar a partir do modo e da necessidade de como ele é empregado, que pode ser voltado para dados ou para arquitetura [3]. Caso o *framework* seja voltado para dados o desenvolvimento de aplicações é realizada pelas várias maneiras de combinar as instâncias das classes já existentes.

Já os *frameworks* voltados para a arquitetura possuem subclasses desenvolvidas tendo como base as classes pré-definidas do *framework*. O mesmo autor define as seguintes categorias:

- *Framework* de Infraestrutura do Sistema: simplificam o desenvolvimento da infra-estrutura de sistemas portáveis e eficientes, como por exemplo, os sistemas operacionais, sistemas de comunicação, interfaces com o usuário e ferramentas de processamento de linguagem. Em geral são usados internamente em uma organização de *software* e não são vendidos a clientes diretamente.

- *Framework* de integração de *middleware*: são usados, em geral, para integrar aplicações e componentes distribuídos. Eles são projetados para melhorar a habilidade dos desenvolvedores em modularizar, reutilizar e estender sua infra-estrutura de *software* para funcionar em um ambiente distribuído. Exemplos dessa classe de *framework* são o "*Object Request Broker*" (ORB), *middleware* orientado a mensagens e bases de dados transacionais.

- *Framework* de Aplicações Corporativas: este tipo de aplicação exige um grau de confiabilidade, desempenho mais elevado e são empregados no desenvolvimento de aplicações comerciais para usuários finais. Como exemplo, tem-se os *frameworks* para aplicações financeiras ou sistemas contábeis.

A outra classificação dos *frameworks* é quanto à maneira que acontece a extensão. Esta classificação se divide em três partes: *Framework* de Caixa Branca baseia-se nas características da orientação a objetos, como herança e ligação dinâmicas e faz-se necessário aos desenvolvedores um bom conhecimento interno do *framework*; *Framework* de Caixa Preta baseia-se na composição de objetos, ou seja, ele define a interface com os componentes de modo que elas sejam conectadas ao *framework* por meio de composição de objetos, tendo maior facilidade de uso e extensão que os de caixa branca e; *Framework* de Caixa Cinza utiliza as características do *framework* de caixa branca e caixa preta provendo tanto a flexibilidade quanto à capacidade de extensão.

2.2.2 Vantagens e desvantagens de Frameworks

Para Maia, um *framework* ao possuir as características (herança, polimorfismo, abstração de dados e etc) das linguagens de programação orientadas a objetos pode facilitar a reusabilidade [32]. Já outras vantagens apontadas por este mesmo autor são: Diminuição das linhas de código em um módulo do *framework*; Possibilidade de utilizar outras técnicas de reuso em conjunto, por exemplo, padrões de projetos (*design patterns*) e componentes; Diminuição dos erros no código já que ele é usado em várias aplicações e; Aumento da qualidade de *software*, facilidade na manutenção, pois quando um erro é corrigido no *framework*, automaticamente, ele é corrigido nas aplicações desenvolvidas a partir deste *framework*.

Algumas desvantagens são apontadas por Albuquerque como, dificuldade em desenvolver um *framework*, considerando, por exemplo, para qual plataforma o *framework* vai ser construído e quais suas limitações. Caso o *framework* não possua uma documentação apropriada, certamente ele não será bem utilizado e; O processo de depuração de erro pode ser complicado porque é difícil distinguir quando o erro é do *framework* ou da aplicação que o utiliza. Caso o erro esteja no *framework* pode ser impossível o usuário conseguir corrigi-lo [3].

2.3 Robótica

A robótica é ciência que estuda a construção de robôs. Ela envolve várias outras disciplinas como engenharia mecânica e elétrica, inteligência artificial, engenharia eletrônica, física entre outras. Para Secchi, a origem da palavra “robô” vem da palavra tcheca robota que significa “trabalho forçado” [50].

Na década de sessenta foram introduzidos na indústria, de modo significativo, os robôs manipuladores como um elemento a mais do processo produtivo. Essa proliferação, motivada pela ampla gama de possibilidades que oferecia, despertou o interesse dos pesquisadores em conseguir manipuladores mais rápidos, precisos e fáceis de programar. A conseqüência direta desse avanço originou um novo passo na automação industrial, que tornou mais flexível a fabricação robotizada.

Atualmente grandes avanços na robótica fazem com que ela se apresente como algo cotidiano nas vidas dessa geração [42]. Assim, é comum ver robôs industriais que soldam,

pintam e movimentam grandes peças. Robôs que atuam em laboratórios farmacêuticos, em salas de cirúrgicas ou nas atividades diárias de pessoas necessitadas de atenções especiais.

Assim, a robótica se apresenta em diversas aplicações para o homem contemporâneo, uma das mais conhecidas é a aplicação industrial, mas os robôs podem ser uitilizados para uma vasta gama de finalidades, como entretenimento (ex: brinquedos, atores, monstros de filmes), realização de ações à distância, exploração de ambientes insalubres e recentemente na educação.

2.3.1 Robótica Educativa

A robótica na educação migra com os computadores no âmbito escolar. Este surge nos anos 70, inicialmente nos Estados Unidos e só a partir de 1980 começam a ser inseridos no Brasil. De acordo com Papert as primeiras experiências com robôs nas instituições educacionais objetivavam a realização de atividades de programação, o resultado foi um novo leque de oportunidades pedagógicas [39]. Assim, existem diversos projetos de robótica para educação. Dentre os mais conhecidos estão os projeto LEGO, os projetos VEX e projeto Robotino FESTO® Ltda. LEGO Mindstorms NXT é uma linha do brinquedo LEGO lançada comercialmente em 2006, voltada para a educação. No dia 4 de janeiro de 2006, na feira *Consumer Electronics Show em Las Vegas*, nos Estados Unidos da América, a LEGO apresentou ao público a nova geração do *Mindstorms* [27].

O *Mindstorms* NXT é equipado com um processador mais potente, *software* próprio e sensores de luz, de toque e de som, permitindo a criação, programação e montagem de robôs com noções de distância, capazes de reagir a movimentos, ruídos e cores, e de executar movimentos com razoável grau de precisão. Os novos modelos permitem que se criem não apenas estruturas, mas também comportamentos, permitindo a construção de modelos interativos, com os quais se aprendem conceitos básicos de ciência e de engenharia [24].

Já os projetos da plataforma VEX, muito conhecida pelo evento que organiza (VRC – VEX *Robotics Competition*), oferecem uma grande variedade de produtos e tecnologias, que podem se adaptar para alunos desde o Ensino Fundamental I até o Ensino Superior. Para os mais novos, oferece uma plataforma de fácil acesso, com programação simplificada por uma interface gráfica e plataforma segura para montagem, enquanto que, para os mais experientes, oferece produtos ideais para aplicações robustas, com robôs que

podem se adaptar a situações de stress mecânico e a movimentos complexos através das diversas combinações de estruturas que os kits VEX oferecem [58].

O projeto Robotino, é um robô móvel desenvolvido pela empresa FESTO® Ltda, (Figura 2.7), com o propósito didático de oferecer ao usuário um primeiro contato com uma tecnologia que pode ser aplicada principalmente na área de automação industrial. O robô é composto por três rodas *Mecanum omnidirecionais* dispostas em ângulos de 120°, os quais são individualmente controláveis, Processador PC104 300MHz, OS Linux, Cartão de memória flash 256MB, Ponto de acesso a rede wireless LAN.

O Robotino possui uma diâmetro (d) igual a 370mm e altura (h) igual a 210mm, sensores de distância (por infravermelho), sensor de choque (também chamado de *Bumper*), sensor de cor, sensor indutivo, sensor de imagem (uma câmera com resolução VGA), *encoders* ópticos rodas, medição de energia para todo o sistema e os diversos motores, bem como um monitor de voltagem da bateria [10].

Além disso, pode ser equipado com um scanner a laser, um giroscópio e um sistema de localização em espaços interiores (*Northstar* pela evolução da robótica). Ele tem várias interfaces: USB, *Ethernet*, 8 entradas digitais e 8 analógicas, 8 saídas digitais, potência adicional do motor para a condução de cargas elevadas, e entrada de adicional de *encoder* [10].

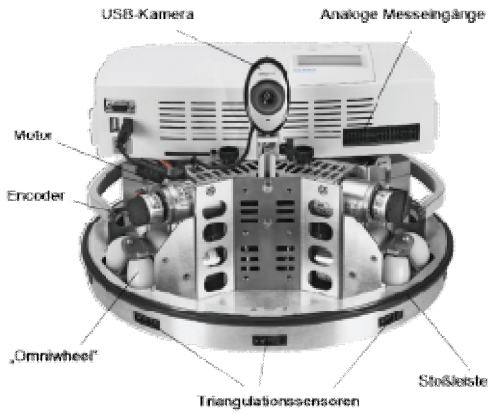


Figura 2.7 - Robotino. Fonte:(FESTO® Ltda, 2015).

2.4 Método da Aprendizagem Cooperativa e Competitiva

A ideia de cooperar e de competir entre alunos, não é nova. Observem algumas definições apresentadas por alguns autores sobre o que entendem ser a aprendizagem cooperativa e competitiva.

Para Hsieh e seus colegas, a aprendizagem cooperativa é um método de ensino em que os alunos trabalham em conjunto com o objetivo de maximizarem a sua própria aprendizagem e a dos colegas [17]. Na mesma linha, Jin define a aprendizagem cooperativa como o trabalho em grupo devidamente estruturado, de modo que todos os alunos interajam, troquem informações e possam ser avaliados individualmente pelo seu desempenho [21].

A aprendizagem cooperativa surge da necessidade de inserir metodologias interativas entre os alunos ou usuários, em conjunto com o professor para que estabeleçam buscas, compreensão e interpretação da informação. Esta técnica exige que dois ou mais alunos trabalhem juntos visando atingir um objetivo específico.

Para Webster, a competição é a interação de indivíduos da mesma espécie ou espécies diferentes (humana, animal ou vegetal) que disputam algo. Esta disputa pode ser pelo alimento, pelo território, pela luminosidade, pelo emprego, pela fêmea, pelo macho, etc. Logo, a competição pode ser entre a mesma espécie ou de espécie diferente. Na educação a aprendizagem competitiva acontece através de competições entre os alunos ou entre equipes de alunos [61].

Para Regueras e seus colegas, a aprendizagem competitiva acontece quando dois ou mais alunos estão envolvidas onde normalmente só um ou alguns participantes vão ganhar e outros não. Existe concorrência quando há escassez de um resultado desejado. Os indivíduos e/ou grupos estão posicionados para a realização desse resultado. Por exemplo, no atletismo, duas equipes se envolvem em um esporte com o objetivo de ganhar [29].

Já Arends reconhece que a origem da aprendizagem cooperativa e competitiva remonta à Grécia antiga, embora, defenda que os desenvolvimentos contemporâneos resultem do trabalho de psicólogos educacionais e teóricos da pedagogia no início do século XX [4].

No entanto, vale ressaltar que as vantagens do trabalho em grupo e competitivo, já estariam nos pensamentos de grandes pedagogos europeus do século XIX (Herbart, Froebel, Pestalozzi, Johnson [22]). Também as teorias mais recentes de processamento de

informação e alguns teóricos do desenvolvimento, tais como Piaget e Vygotsky constituem contributos fundamentais na expansão desta prática [40,59].

De acordo com Leland, Ribeiro e seus colegas o sucesso da aprendizagem cooperativa e competitiva tem como base elementos essenciais de cooperação e de competição, que precisam ser cuidadosamente estruturados no processo de implementação do método [28, 44]. Assim, os autores descrevem os seguintes elementos da aprendizagem cooperativa:

- Interdependência positiva: cabe ao professor estabelecer uma tarefa clara e um objetivo a ser atingindo pelo grupo, de modo, que os alunos acreditem que "ou afundam todos, ou nadam todos juntos". A interdependência positiva existe quando os membros do grupo percebem que estão ligados uns com os outros de uma forma que não se pode ter sucesso a menos que todos o consigam. Dessa forma, todos os membros aprendem a valorizar o esforço de cada colega [44].
- Responsabilidade individual e de grupo: o grupo deve ser responsável pelo sucesso dos seus objetivos. Cada membro deve ser responsável por contribuir com a sua parte do trabalho, garantindo desse modo a participação efetiva de todos, evitando que se apóiem somente no trabalho realizado pelos colegas. Os objetivos devem ser claros e todos os membros do grupo deverão ser capazes de medir o seu progresso, esforço e contribuição individual. Assim, considera-se que existe responsabilidade individual, quando o desempenho de cada aluno é avaliado e os resultados são fornecidos ao grupo. Isso verifica quem necessita de mais assistência, apoio e estímulo para completar a tarefa. Um dos objetivos dos grupos de aprendizagem cooperativa é contribuir para que cada membro torne-se uma pessoa mais forte no exercício dos seus direitos. Os alunos aprendem em conjunto, para posteriormente, serem capazes de evoluir individualmente [24].
- Interação promotora, de preferência face a face. Esta interação ocorre quando os membros compartilham recursos e aprendem a incentivar os membros da equipe. Os grupos de aprendizagem cooperativa são tanto um sistema de apoio acadêmico (cada aluno tem alguém que está empenhado em ajudá-lo a aprender) quanto um sistema de apoio pessoal (cada aluno tem alguém que está comprometido com ele, como pessoa) [44].

O último elemento da aprendizagem cooperativa é ensinar aos alunos as competências interpessoais e de grupo. Eles são obrigados a aprender conceitos acadêmicos, mas também pequenas regras de comportamentos pessoais [24].

Agora os autores definem os elementos para aprendizagem competitiva:

O primeiro elemento é ajudar os alunos a ter consciência de seus sentimentos competitivos no contexto da baixa ou perda; O segundo é ajudar os alunos a testar a sua capacidade de permanecer consciente em situações competitivas e o último é fomentar aos alunos que a equipe vencedora ganhará a aprovação do professor na disciplina.

Para Lin seus amigos, aprendizagem cooperativa é uma estratégia de ensino que pode ajudar os alunos a atingirem o melhor efeito de aprendizado como, estimular a criatividade, aprender com responsabilidade e despertar habilidades sociais e comunicativas nos alunos [31]. Assim, adaptação do *framework* proposto ao método de aprendizagem cooperativa e competitiva pode ajudar o estudante a eliminar suas dúvidas durante os experimentos das atividades.

2.5 Usabilidade

A usabilidade facilita com que as pessoas podem empregar uma ferramenta ou objeto a fim de realizar uma tarefa específica e importante. A usabilidade pode também se referir aos métodos de mensuração da usabilidade e ao estudo dos princípios por trás da eficiência percebida de um objeto [25].

A norma ISO 9241-171, define que a usabilidade pode ser mensurada em termos da eficácia, eficiência e satisfação; quando usuários específicos usam um produto ou sistema para atingir objetivos específicos, em um contexto específico de uso [20]. Contudo, um produto ou sistema que não pode ser usado para alcançar os objetivos de tarefas nunca vai ser eficiente e, portanto, não é utilizável ou acessível. À medida que o nível de eficácia é aperfeiçoado, os níveis de eficiência e satisfação são atingidos.

Esta dissertação de mestrado utilizará dos métodos da usabilidade para avaliar o framework proposto. A escolha das técnicas e ferramentas apropriadas depende da etapa do desenvolvimento do *software*, ou seja, para este caso em que a aplicação esta sendo implantada no ambiente de sala de aula, buscou-se por meio do aluno, a certificação que a solução prestar-se-á para os fins que motivaram a sua aquisição.

Para Cybis e seus colegas, o termo teste de usabilidade é frequentemente utilizado indiscriminadamente para se referir a qualquer técnica usada para avaliar um produto ou sistema [7]. Em outras palavras, é uma das técnicas de avaliação existentes para aferir a qualidade de uso de *softwares*. Nesta abordagem utiliza-se o Teste de Validação de Usabilidade. Ele acontece numa fase mais avançada do ciclo de desenvolvimento e verifica

como o produto se enquadra em relação a padrões de usabilidade, de desempenho, de históricos definidos no começo do projeto e do dialogo entre o usuário e a interface do *software*. Na próxima subseção descrevem-se especificamente as técnicas de avaliação para esta pesquisa.

2.5.1 Técnicas de Avaliação de Usabilidade

Para Cybis as técnicas de Avaliação de Usabilidade podem ser classificadas quanto ao seu objetivo. No entanto e, para este caso, daremos ênfase à avaliação prospectiva, por meio dos questionários. Para isso, é necessário empregar critérios para medir não só a usabilidade em um contexto de uso genérico, mas também levar em consideração o projeto pedagógico da aprendizagem que está inserido na aplicação. Assim, foram encontrados vários questionários [57,25, 26, 35, 38, 1], porém, consideraram-se apenas duas classes para este projeto:

a) Usabilidade Geral: baseada em Nielsen e nas questões do modelo de questionário ISO 9241/10 [19, 35] e,
b) Usabilidade Pedagógica: baseada em Nokelainen [37].

O questionário de Nielsen consistiu em dez heurísticas que aplicada à usabilidade geral e voltada para a usabilidade de produtos [35]. Seus princípios são: Visibilidade do estado ou contexto atual do sistema - o sistema deve orientar e conduzir o usuário, informando sobre o que está acontecendo, por meio de realimentação apropriada, em tempo razoável;

Compatibilidade com o mundo real - o sistema deve adotar uma terminologia familiar ao usuário, exibindo informações em ordem lógica, natural e coerente com o modelo mental do usuário; Controle e liberdade do usuário - o usuário deve manter o controle sobre o processamento de suas ações, com a opção de desfazer e refazer operações;

Consistência e padrões - os usuários devem ser poupados de ter que deduzir quais termos, situações e ações têm significados semelhantes. Prevenção de erros - o projeto da interface deve prevenir a ocorrência de erros e ajudar a corrigí-los, caso ocorram; Reconhecimento ao invés de memorização - as instruções para uso do sistema devem estar facilmente disponíveis para consulta;

Flexibilidade e eficiência de uso - a interface com o usuário deve adaptar-se ao contexto, e às necessidades do usuário, promovendo a eficiência de uso; Projeto estético minimalista - a interface com o usuário deve ser simples, e as informações devem ser fornecidas ao usuário na medida em que se façam necessárias;

Diagnosticar e corrigir erros - o sistema deve oferecer suporte aos usuários na identificação de problemas. As mensagens de erros devem ser claras, indicando precisamente o problema e sugerindo soluções; Informações de ajuda e documentação - a documentação do sistema deve ser fácil de pesquisar, focada nas tarefas e estar sempre disponível.

A usabilidade pedagógica tem a finalidade de satisfazer as necessidades dos alunos para a realização de tarefas propostas por meio de componentes da interface do material de aprendizagem. Conforme a classificação de Nielsen, a usabilidade pedagógica é um sub-conceito de utilidade [35]. Para Nokelainen, a usabilidade pedagógica deve prover um projeto do material de aprendizagem cujas funções facilitam a aprendizagem deste material e sua distribuição [37].

2.6 Conclusão

Neste capítulo foram apresentados os fundamentos necessários para o desenvolvimento da proposta desse projeto. Explanaram-se sobre linguagens de programação, *Frameworks* bem como seus métodos e técnicas. Apresentaram-se o conceito de Robótica na Educação, métodos de Aprendizagem Cooperativa/Competitiva e finalizou-se com as Técnicas de Usabilidades.

A explanação sobre as linguagens de programação foi de extrema relevância neste contexto, pois, nos próximos capítulos utilizaremos uma Linguagem de programação para compor o *framework* proposto. Já a explicação sobre *frameworks***,** visou entender as características necessárias para desenvolver o aplicativo que possibilitará a reutilização de códigos na implementação do robô.

O esclarecimento da Robótica Educacional apresenta-se neste trabalho como um mecanismo, que possa auxiliar as aulas práticas de linguagens de programação por meio de um robô móvel. Além disso, apresentou-se a metodologia de aprendizagem cooperativa e competitiva norteando nos próximos capítulos, pôr em prática os conteúdos compreendidos

pelos alunos por meio do *framework* proposto. Essa metodologia poderá permitir a socialização e a motivação entre os alunos de forma genuína e singular.

As técnicas de usabilidade visam verificar futuramente a satisfação do estudante (em relação ao produto) por meio da usabilidade geral e pedagógica, ou seja, se o *framework* proposto no ambiente de sala de aula faz o que promete. Assim, acredita-se que um *framework* auxiliado pela robótica e adaptada ao método de aprendizagem cooperativo e competitivo poderá auxiliar de forma reflexiva o ensino de Linguagens de Programação. Dessa forma, descreveram-se as teorias e conceitos fundamentais para que possamos utilizá-los como base na construção do *framework* proposto.

Capítulo 3- Trabalhos Relacionados

O objetivo deste capítulo é apresentar uma combinação de temas que estão distribuídos em diversos trabalhos científicos visando identificar elementos que possam contribuir com o desenvolvimento deste trabalho de pesquisa. Os temas aqui abordados são *Framework* e Linguagens de Programação, *Framework* e Robótica e, *Framework* e Aprendizagem Cooperativa/Competitiva.

Na seção *Framework* e Ensino de Linguagens de Programação descrevemos alguns projetos com o foco no desenvolvimento do raciocínio lógico. Já em *Framework* e Robótica apresentamos alguns projetos utilizando *framework* e robótica no processo de ensino-aprendizagem. Na seção *Framework* e Aprendizagem Cooperativa e Competitiva cometamos alguns projetos aplicando *frameworks* ao método de aprendizagem.

Assim, propõem-se as discussões dos trabalhos relacionados em função de tais temas, apontando sempre o que foi feito em cada trabalho. Por fim é apresentado um quadro contendo todos os trabalhos abordados com itens comuns e divergentes a esta dissertação de mestrado.

3.1 Framework e Ensino de Linguagem de Programação

Fleischfresser demonstra em seu artigo um *framework* desenvolvido para a disciplina de Mecânica Vetorial, mais especificamente à cinemática de corpos rígidos interconectados. O projeto envolveu acadêmicos do 2º ano de engenharia, onde o professor disponibilizou o código Fortran inicial. Os alunos ficaram encarregados de entender a lógica computacional utilizada e relacionar os conceitos e aplicações vistos em sala de aula [11].

O *framework* relativo tem como objetivo desenvolver o raciocínio crítico dos estudantes, além de ser um bom exemplo de aprendizagem ativa nas aulas de engenharia. Os resultados não só melhoraram a receptividade dos alunos como também motivaram as atividades colaborativas entre eles [11]. Porém, o autor não utiliza nenhum mecanismo para auxiliar sua metodologia na parte prática.

O projeto *Scratch* apresenta um *framework* denominado de *Scratch* que permite criar animações, histórias interativas e até mesmo jogos por meio de uma linguagem de programação simples. A interface pode ser acessada por qualquer pessoa que tenha o mínimo de conhecimento sobre desenvolvimento de *software*. As telas disponibilizadas pelo aplicativo permitem que você desenhe livremente ou que utilize as ilustrações já existentes no banco de dados [49]. O objetivo do *framework* é escrever frases e criar objetos básicos, como quadrados, círculos e triângulos. Os resultados demonstram que o *Scratch* é uma excelente ferramenta para as disciplinas de lógica de programação [33]. No entanto, esse projeto não relata a presença de um robô ou qualquer outra ferramenta práticas como apoio em sua metodologia.

Medeiros e seus colegas apresentam um *framework* para criação de jogos voltados para o ensino de lógica de programação. A estrutura foi desenvolvida por meio da integração do ambiente visual de programação *Blockly* com cenários criados em HTML5 [34]. O objetivo dos autores é estimular o raciocínio lógico em crianças do nível básico visando desenvolver uma engrenagem motivadora para o processo de aprendizagem. O resultado é direcionado para trabalhos futuros, pois os autores pretendem desenvolver jogos com uma estrutura e qualidade desafiadora. Em suma, os autores apresentam um *framework* para o ensino da lógica de programação, porém não comentam qual metodologia utilizam para a fixação dos conteúdos por meio prático.

No trabalho de Fontes e Silva são apresentados os resultados de um estudo de campo sobre textos usados no ensino da disciplina linguagens de programação, em algumas escolas técnicas. Foi investigado o tipo de material didático adotado na disciplina, os critérios usados na escolha, as principais características do material, e a viabilidade de uma proposta alternativa para utilização desse material [9]. A pesquisa de campo foi desenvolvida em uma escola técnica de ensino médio, ao longo de dois meses, obtendo-se um desempenho superior dos alunos submetidos à estratégia alternativa. Os resultados demonstram um aproveitamento de 90,30% para o planejamento de aulas de linguagens de programação, bem como para o planejamento curricular de escolas de ensino médio voltadas à formação de profissionais de computação [9].

3.2 Framework e Robótica

A utilização de *framework* no auxílio à educação tem se mostrado útil no processo de ensino-aprendizagem. Por exemplo, pesquisadores como Sumita e seus colegas relatam que o Departamento de Segurança Interna dos EUA emprega engenheiros e graduados de computação, porém eles enfrentam dificuldades com a Infra-Estrutura de Proteção Críticas (IPC) [54]. Este *framework* está relacionado com tratamento de problemas de projetos e a maioria dos currículos de graduação não incorpora IPC. Dessa forma, os autores propõem uma estrutura curricular flexível para integrar IPC no ensino de graduação por meio dos módulos interdisciplinares IPC auto-suficiente. O módulo do curso é uma unidade curricular distinta a cursos já existentes e sem necessidade de mudança pedagógica.

A estrutura foi projetada para ser utilizada em várias disciplinas e os módulos são projetados para atuar em diferentes níveis de experiência de graduação. Os autores reconhecem que os alunos podem trabalhar em setores de negócios voltados para questões de proteção de infra-estrutura. Dessa forma, uma equipe transdisciplinar do corpo docente desenvolveu uma estratégia para garantir que os alunos implementasse as questões IPC.

Assim, o objetivo desses autores é utilizar o *framework* IPC e preparar os estudantes para resolução de problemas nas indústrias como, questões de design, implementação e manutenção de ativos de infra-estruturas robustas e sustentáveis. Os autores finalizam seu artigo enfatizando em seus experimentos, que a utilização do *Framework* IPC incorporado ao currículo de graduação em engenharia, foi mais eficaz do que o uso de métodos tradicionais e que pode preparar os estudantes para resolução de problemas de projetos no âmbito industrial.

Observou-se que no projeto do *framework* IPC, os autores não comentam sobre a adaptação do *framework* a métodos pedagógicos e não utiliza robôs móveis como mecanismo de apoio. Porém, desenvolveram e adaptaram um *framework* IPC a várias disciplinas. Além disso, a construção do conhecimento é adquirida por meio de resolução de problemas.

Já autores como Frank e seus associados relatam que uma das principais mudanças nos currículos de engenharia, tem sido a ênfase no desenvolvimento de competências pessoais. Essas habilidades incluem a capacidade de aprender por meio da colaboração e através da reflexão crítica do próprio indivíduo ou grupo. Esta pesquisa relata sobre como aprendizagem cooperativa foi integrada na prática ao módulo de engenharia para calouros,

e explora as experiências de estudantes ao serem encorajados a participar da reflexão significativa [11].

Em particular, os autores detalham em seus experimentos que os módulos de apresentações e dos fenômenos que influenciam as atitudes dos estudantes de engenharia em relação à aprendizagem cooperativa e reflexiva, resultam em uma melhor aprendizagem. Os experimentos são implementados por meio de relatórios, ou seja, das próprias palavras dos alunos e sobre sua compreensão da aprendizagem cooperativa. Em outras palavras, os alunos demonstram que foi possível identificar os elementos-chave que são essenciais para o sucesso do grupo.

Por exemplo, observe a descrição do relato do aluno 1: "Tendo completado este projeto, agora tenho uma melhor compreensão dos diferentes tipos de situações de engenharia. Eu também identifiquei que a combinação de teoria e prática em um projeto foi um desafio muito interessante, pois me mostrou como as peças de *hardware* e *software* estão estreitamente ligadas, tanto na forma, como no código e nos componentes utilizados para fazer tarefas, ou seja, para detectar os movimentos do produto ao redor da pista. Coletivamente, acredito que o grupo desenvolveu algumas habilidades interessantes para a vida profissional e isso foi gratificante " [12].

Assim, o projeto da integração do método cooperativo ao módulo de engenharia, motiva os alunos e, a fixação do conteúdo é por meio da socialização entre os estudantes. Porém, os autores não comentam sobre a adaptação de *framework* e nem de robô móvel para fixar os conceitos.

Para Hart e seus colegas apresentam um *framework* em andamento denominado de *Robot Task Commander* (RTC), ele é uma estrutura para programação de robô e pode ser utilizado em diferentes contextos. O RTC foi criado pela NASA-JSC em conjunto com a *General Motors* para uso com o *Valkyrie* (plataformas robô humanóide). O RTC fornece uma IDE apropriada para programação de robôs com vantagem sobre outros *frameworks* de programação. Os autores informam que há vários níveis de flexibilidade para os desenvolvedores.

O objetivos do RTC são de implementar os seguintes princípios: Aplicações para robôs; Controlar o fluxo do robô; Unidade de cálculo deve ser generalizada e, portanto, deve ser escrito em termos de interfaces abstratas (ou seja, tipos de dados); Toda a funcionalidade deve ser armazenada em uma biblioteca acessível para reutilização e Interfaces adequadas devem existir para os especialistas e não-especialistas. Em resumo, RTC é um *framework* para desenvolvimento de aplicações robóticas. A capacidade de

integrar essa estrutura em aplicações faz o RTC ter uma contribuição única e adequada para resolver a próxima geração de tarefas robóticas. Os experimentos de eficácias do *framework* são inferidos quando se programa o robô *Valkyrie*. Os resultados mostram que esta estrutura tem um domínio singular sobre os objetos executados pelo robô. No entanto, observou-se que não utilizam métodos pedagógicos em seu projeto [55].

Outras pesquisas apresentadas por Wan-Ling e Selma Šabanović utilizam um *framework* para analisar qualitativamente dados coletados por meio da observação (pessoal, residentes, visitantes) e da interação homem-robô (PARO) em um lar de idosos. A pesquisa teve como objetivo compreender como os usuários em sociedade utilizariam um robô na interação diária.

O *framework* da formação social tem como finalidade identificar fatores sociais que apoiarão a interação de robôs em asilos de idosos e em ambiente familiar com flexibilidade, ou seja, como os robôs podem ser utilizados e integrados em ambiente familiar e instituições sociais visando seus aperfeiçoamentos no ambiente real.

Os resultados e aplicação da formação social sugerem que a unidade de análise da pesquisa deve ser expandida visando incluir outros usuários em ambiente sociais [60]. Em suma, o projeto do *framework* voltado para analisar qualitativamente dados coletados por meio da observação e da interação homem-robô não integra métodos pedagógicos. No entanto, utilizam um robô móvel para exemplificar os conceitos práticos.

Autores como Nikolaidis e seus associados apresentam um modelo de *framework* para a aprendizagem automática. A ideia dessa proposta foi do usuário executar ação das mãos em conjunto com o robô. O objetivo dos autores é permitir que o robô desenvolva tarefas de colaboração com um ser humano. Os resultados indicam que esse tipo de *framework* em ação conjunta com o robô pode executar tarefas colaborativas nas indústrias e escolas [36]. Entretanto os autores não comentam a presença de metodologias pedagógicas visando motivá-los no processo de ensino-aprendizagem.

3.3 Framework e Aprendizagem Cooperativa e Competitiva

Trabalho semelhante apresentado por Yussiff, Ahmad e Oxley mostra que a associação de *frameworks* ao ambiente e-*collaboration* por meio de mídia social é uma abordagem pedagógica promissora. Os autores têm como objetivo a integração da tecnologia com a grade curricular no ensino superior em um único módulo didático. Os

resultados não são demonstrados porque os autores ainda não desenvolveram um protótipo empiricamente testado. O teste será utilizado quando o ensino colaborativo *on-line* for desenvolvido em uma plataforma portável e sua usabilidade será confirmada por meio de uma série de avaliações [67]. Em suma, os autores desenvolveram uma estrutura de *framework* para adaptá-lo ao currículo do ensino superior auxiliada pelo método da aprendizagem cooperativa, mas não comentam qual a forma de construção do conhecimento.

Pesquisadores como Krithivasan e seus colegas relatam que a aprendizagem competitiva inserida na disciplina de robótica entre alunos, proporciona experiências práticas. O objetivo dos autores é implementar uma aprendizagem teórica de forma competitiva. Os resultados enfatizam que mais de 95% das equipes aplicaram seus conhecimentos teóricos, em sistemas embarcados e 60% das equipes aplicaram seus entendimentos abstratos para desenvolver soluções de determinados problemas. Além disso, mais de 30% das equipes estavam aptas a analisar criticamente o problema de forma eficaz [24].

Os resultados preliminares afirmam que a metodologia utilizada é atraente e motivadora, pois eles utilizam o método de aprendizagem competitiva na disciplina de robótica para resolução de problemas [24]. No entanto, o projeto não utiliza qualquer tipo de *framework.*

3.4 Concepção

Os temas aqui explanados foram de extremas relevâncias para essa dissertação, porque eles contêm os critérios básicos para a construção do *framework* proposto. Na seção 3.1, comentou-se sobre os *frameworks* aplicados na disciplina de linguagem de programação. Nesta seção os *frameworks* apresentados sempre tratam de problemas de raciocínio lógico, desenvolvimento cognitivo e dificuldades de compreensão.

Na seção 3.2 explanou-se sobre *framework* no processo de ensino-aprendizagem auxiliado pela robótica. Os projetos apresentados nesta seção utilizam *frameworks* para manipular um robô. A ideia central destes projetos é utilizar a robótica para consolidar os conteúdos ministrados em sala de aula por meio da prática. Já na seção 3.3 comentou-se sobre *frameworks* aplicados ao método de aprendizagem cooperativa e competitiva. Os autores destes projetos demonstram que a aprendizagem cooperativa possibilita aos

estudantes a motivação e a socialização de conhecimentos. Igualmente acontece com a aprendizagem competitiva.

Linguagens de programação envolvem conceitos complicados para a compreensão do aluno recém chegado à universidade. Dessa forma, a hipótese dessa dissertação norteia desenvolver uma ferramenta para manipular um robô na disciplina de linguagem de programação visando proporcionar ao aluno, itens como, perceber, aprender, recordar e pensar.

Assim, os critérios necessários para o desenvolvimento desta ferramenta são de desenvolver um *framework* para a disciplina de linguagem de programação e utilizar um robô móvel como apoio didático visando às construções dos conceitos por meio do método de aprendizagem cooperativa e competitiva.

O Quadro 3.1 contém os trabalhos relacionados, os pontos em comum e as divergências com o trabalho proposto. No projeto de Sumita e seus colegas, os itens c, d, e e são comuns a este projeto, porém, o item a e b divergem com esta proposta. No projeto de Frank e seus associados, os itens b, c e d são comuns a este projeto. Neste projeto a integração do método cooperativa ao módulo de engenharia motiva os alunos e a fixação do conteúdo é adquirida por meio da socialização entre os estudantes. Porém, os itens a e e) divergem com esta proposta.

No projeto de Wan-Ling e Selma, as opções c e e são comuns a esta dissertação. No entanto, os itens a, b e d são divergentes. No projeto de Nikolaidis e seus companheiros, os itens comuns são c e e. Os divergentes são a, b e c. No trabalho de pesquisa de Yussiff e seus associados, os pontos em comum a esta dissertação são a, b e d. As opções divergentes são c e e. No artigo de Krithivasan e seus colegas, a divergência está no item a. Os pontos semelhança são b, c, d e e. Já para o projeto desenvolvido por Fleischfresser as opções a, b, c, e d são comuns a esta dissertação. No entanto, o item e é divergente a esta proposta. No projeto Scratch os itens a, b, c e d são comuns a este projeto. No entanto, ele diverge no item e. Em Medeiros os itens em comum são a, b, c e d e se diverge em e. No projeto de Clébe e seus associados, as opções a, b, c e d são comuns a esta dissertação e o divergente está no item a.

Quadro 3.1 - Comparação geral do trabalhos relacionados com o trabalho proposto

Nº	Trabalhos	a)	b)	c)	d)	e)
1	Sumita *et al.* [54]	N	N	S	S	S
2	Frank *et al.* [12]	N	S	S	S	N
3	Stephen *et al.*[55]	N	N	S	S	S
4	Wan-Ling e Selma [60]	N	N	S	N	S
5	Nikolaidis *et al.* [36]	N	N	S	N	S
6	Yussiff *et al.* [63]	N	S	N	S	N
7	Krithivasan *et al* [24]	N	S	S	S	S
8	Fleischfresser [11]	S	S	S	S	N
9	Scratch [47]	S	S	S	S	N
10	Medeiros et al. [34]	S	S	S	S	N
11	Cléber *el al.*[9]	S	S	S	S	N
12	Proposta	S	S	S	S	S

Legenda do Quadro 3.1

a) Desenvolve e adapta estrutura de *framework* na disciplina de linguagem de programação?
b) Utiliza algum tipo de método de Aprendizagens?
c) Qual a forma de construção do conhecimento (teórico, prático ou teórico-prático)?
d) Prepara os estudantes para resolução de problemas de projetos de engenharia?
e) Utiliza algum mecanismo (robô) como auxilio didático?

3.5 Conclusão

Neste capítulo, discutimos sobre projetos envolvendo *frameworks*, linguagem de programação, robótica educativa e a aprendizagem cooperativa-competitiva. Esses temas foram importantes porque eles possibilitaram a realização de uma análise das diferentes propriedades de cada uma das abordagens dadas pelos autores e posteriormente apresentando-as em quadros comparativas.

O quadro facilitou na escolha das características que farão parte da proposta deste trabalho, ou seja, buscou-se construir uma proposta que incorpore as várias características importantes em uma mesma proposta. A ideia é facilitar a aprendizagem do aluno por meio de aulas práticas nós laboratório. Dessa forma, os critérios definidos para o *framework* proposto foram *framework* aplicado no ensino de linguagem de programa, *framework* auxiliado pela robótica e *framework* adaptado ao método de aprendizagem cooperativa e competitiva. Assim, esses critérios proporcionarão ao próximo capítulo a construção da concepção da arquitetura do *framework* proposto.

Capítulo 4- Concepção da Solução Desejada

Este capítulo apresenta a concepção do *framework* proposto auxiliado pelo método de aprendizagem cooperativa e competitiva via robô móvel. Contudo, descrevemos um conjunto de atividades relacionadas que levam à produção do arcabouço do *framework*. Além disso, utilizamos o modelo orientado a reúso, conforme Sommerville, o estágio de desenvolvimento é semelhante a outros modelos de processo de *software* [53].

Dessa forma, apresentamos as seguintes atividades a serem desenvolvidas como, especificação de requisitos que visa definir elementos que satisfaça as reais necessidades do *framework*. Análise de componentes que norteia selecionar itens de interface gráfica, módulo de comunicação e módulo de controle.

Nas adaptações de requisitos elaboramos uma revisão das atividades, em seguida apresentamos o projeto de arquitetura do *framework*, ela é desenvolvida por meio de um estudo de caso e finalizamos com uma visão geral do modelo do sistema proposto com o objetivo de facilitar o processo da busca de tecnologias que atendam os requisitos necessários para a construção do protótipo.

4.1 Especificação de Requisitos

A especificação de requisitos tem como objetivo satisfazer às reais necessidades do objetivo geral desta pesquisa. Assim, selecionou-se a linguagens de programação C, para ser a linguagem de programação experimental do *framework* proposto, ou seja, utilizaremos C porque é uma das primeiras linguagens de programação compilada de propósito geral, estruturada, imperativa, procedural, padronizada pela ISO e importante para sistemas embarcados [16, 62].

A definição da ementa para o processo de ensino da linguagem de programação C, foi baseada no livro de Herbert (C Completo e Total) e ficou delimitada em conceitos de C, variáveis do tipo inteiro, operadores de aritméticos, operadores de comparação e estrutura condicional [16, 23]. Além disso, utilizou-se esta ementa para definir as características dos

métodos de reuso do *framework* proposto. Já o robô selecionado para esse projeto foi o Robotino da FESTO® Ltda, por ser um robô robusto, ter bibliotecas consistentes, adaptações de placas externas, tem a sua própria rede local e ser bastante resistente a impactos. Vale ressaltar que poderíamos ter escolhido outro robô educacional, por exemplo, o da LEGO, mas, optamos pelo Robotino por ter as qualidades já citadas.

O método de aprendizagem cooperativa é responsável pela organização, pela eliminação de dúvidas e pela socialização das equipes no ambiente de sala de aula. O método de aprendizagem competitiva é responsável pela motivação, interações e competição entre as equipes de alunos em sala de aula.

Em suma, foram definidos os requisitos do *framework* proposto, ou seja, ele deverá auxiliar professor e aluno no ensino de linguagem de programação C com apoio de um robô móvel. O *framework* funcionará inicialmente nas plataformas *Windows* ou Linux e o sistema deve auxiliar a metodologia de aprendizagem cooperativa e competitiva. O *framework* será conectado ao Robotino da Festo *education* e o *framework* deverá ter facilidade na sua condução e deverá ser de fácil aprendizagem. O *framework* aborda apenas conceitos de C, variáveis do tipo inteiro, operadores aritméticos, operadores de comparação e estrutura condicional.

4.2 Análise de Componentes

Dada a especificação de requisitos, realizou-se uma análise de componentes visando auxiliar o professor e o aluno em aulas de linguagem de programação C. Assim, definiu-se que o sistema deve permitir uma estrutura dos conteúdos teóricos; O *framework* deve permitir que o professor insira em formatos de texto e/ou imagens os conteúdos de linguagem de programação C; O *framework* deve permitir que o aluno exercite a teoria por meio do Robotino; O *framework* deve permitir o uso de multimídias e o acesso à internet como apoio didático; O *framework* deve controlar o Robotino para implementar as instruções de linguagem de programação C e o sistema deve manipular variáveis, operadores e estrutura de decisão.

Depois de identificados os critérios, foi elaborada uma análise sobre as correspondências de suas funções para formar e adaptar as estruturas do pacote do *framework* proposto. Nesta atividade foi verificado itens de interface gráfica, módulo de comunicação e módulo de controle.

4.3 Alterações nos Requisitos

Durante esta fase foi elaborada uma revisão dos requisitos das atividades de análise de componentes. Assim, dividiram-se os requisitos em “Pacote Cliente”, que é responsável em controlar os seguintes formatos de texto e/ou imagens: Texto (pdf, doc, docx, Excel, ODF, Whiter e Cal), Imagens (JPG, BMP, GIF e PNG) e interface gráficas em que o usuário tenha facilidade de conduzir as interfaces “Conteúdos”, interface responsável pelo conteúdo básico da disciplina linguagem de programação C; Tela de “Atividade”, interface responsável em implementar as instruções de códigos de linguagem de programação C e executar no Robotino. Tela de “Pesquisa”, interface responsável em oferecer o acesso à internet (somente com conexão a internet esta aba funcionará) e tela de Multimídia, interface responsável em auxilio didático por meio da inserção de áudio, vídeo e documentos.

O Pacote de Comunicação utiliza a arquitetura do tipo Cliente-Servidor, ou seja, conforme Sommerville na arquitetura Cliente-Servidor, a comunicação acontece quando o cliente (professor ou estudante) envia uma instrução pela rede ao processo servidor, e então o processo servidor recebe a instrução, e executa o trabalho solicitado ou procura pelos dados requisitados e envia uma resposta de volta ao cliente, que estava aguardando [53]. Em outras palavras, o usuário utiliza a interface gráfica do *framework* por meio do Pacote de Comunicação para enviar os comandos ao Pacote Servidor, responsável em conter um conjunto de métodos para manipular o Robotino.

4.4 Projeto de Arquitetura do Framework (protótipo)

Neste estágio de atividade o *framework* foi projetado, não para ser definitivo, mas para disponibilizar uma arquitetura reutilizável, visando melhorar os recursos oferecidos. Foi levado em consideração o desenvolvimento dos componentes e como organizá-los para prover o reuso das funcionalidades e classes do *framework* em ambiente pedagógico. Por meio destas atividades foram construídas visões de caso de uso, diagrama de classes e levantamento da arquitetura do *framework*.

4.4.1 Caso de uso

Norteando uma perspectiva de entendimento sobre o *framework* proposto, elaborou-se a documentação do modelo de caso de uso, de acordo com Bezerra, o diagrama de Caso de uso descreve um cenário por meio de sequência de passos com interação entre o usuário e o sistema [5, 30]. No contexto desta dissertação, o caso de uso apresentado na Figura 4.1 mostra as principais funcionalidades do *framework* em relação ao professor e ao aluno, assim temos: **Professor** (Individuo que ministra a disciplina e responsável pelas inserções metodológicas no *framework* proposto); **Aluno** (Individuo que utiliza o *framework* proposto para praticar a teoria).

Assim, o tipo de classificação do *framework* proposto é a mesma do *framework* de caixa cinza, por se tratar do emprego da orientação a objetos e componentes de manipulação de interfaces gráficas.

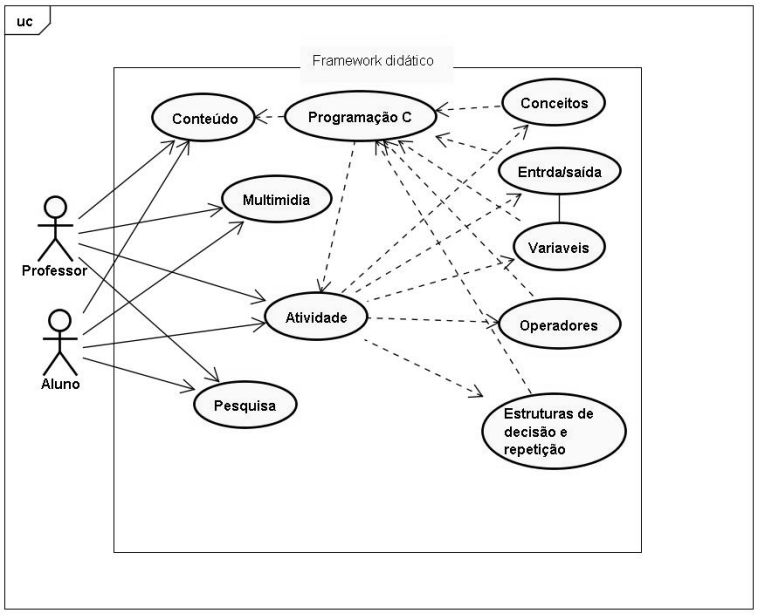


Figura 4.1 - Caso de uso do *framework* proposto

A Figura 4.2 apresenta um diagrama de classe do *framework* proposto. Esse diagrama contém o Pacote Cliente (PC), que contém as seguintes classes: interface, pesquisa, multimídia, atividade e conteúdo. A classe conteúdo chama a classe programação C, ela é responsável em implementar as classes (Alconceito, Al_IO, Variável, Operadores e EstrDR) as quais são responsáveis pelas execuções dos conteúdos de linguagem de programação. Já a classe atividade estar associada à classe programação C. Além disso, a

classe atividade é responsável pela implementação das instruções no Pacote Servidor. O objetivo do Pacote Cliente é de administrar as instruções da disciplina Linguagem de Programação C e enviar para o Pacote Servidor.

Pacote Servidor (PS): O PS contém a classe "controlRobo" é responsável pela implementações dos métodos de comandos das classes de sensor, atuador e palavras reservadas. Este pacote será embarcado no Robotino e pode ser acessado pelo Pacote Cliente. Este Pacote tem como objetivo executar os movimentos do Robotino.

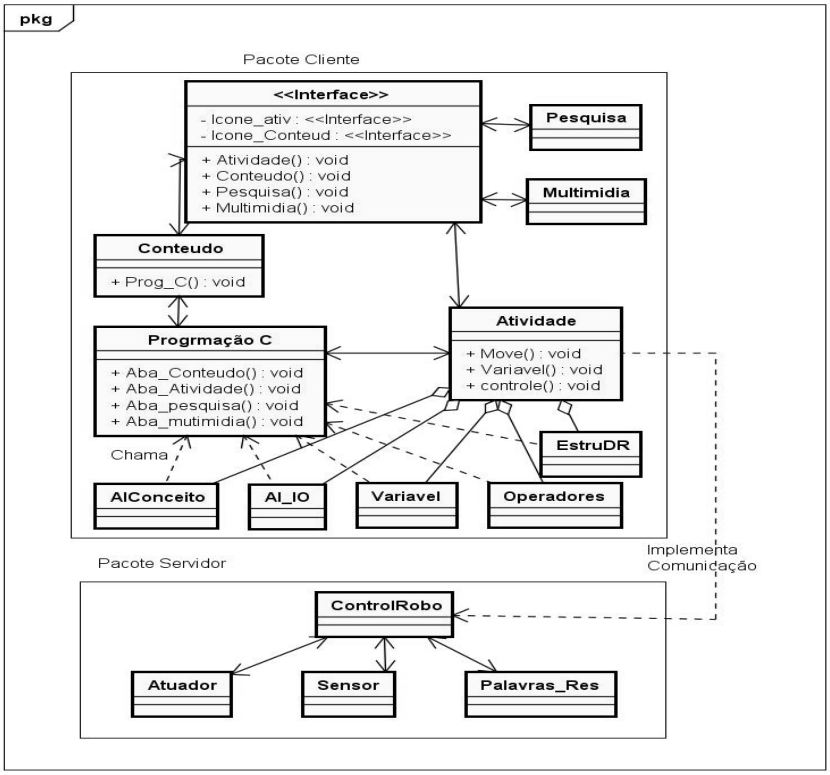


Figura 4.2 - Diagrama de classe do *framework* proposto

Comunicação do *framework* proposto é responsável pelos envios das instruções entre os pacotes (PC e PS). Assim, os processos cliente enviam pedidos para o processo servidor, e este por sua vez implementa os seus pedidos.

A Figura 4.3 apresenta um diagrama de sequência, que tem o objetivo de mostrar como as instruções entre os objetos são trocadas no decorrer do tempo visando à realização de uma atividade ministrada pelo usuário (professor ou aluno). Observe por meio das setas horizontais as atividades do usuário, ou seja, acessa o menu (Pacote Cliente) e em seguida, "Program_C", "Acess_Conceitos" e "AcesAtividades". Após isso, o professor/aluno

solicita uma conexão com o Pacote Servidor. Dessa forma, o PC aceita a conexão e o processo de envio de instrução começa a ser analisado e posteriormente executado.

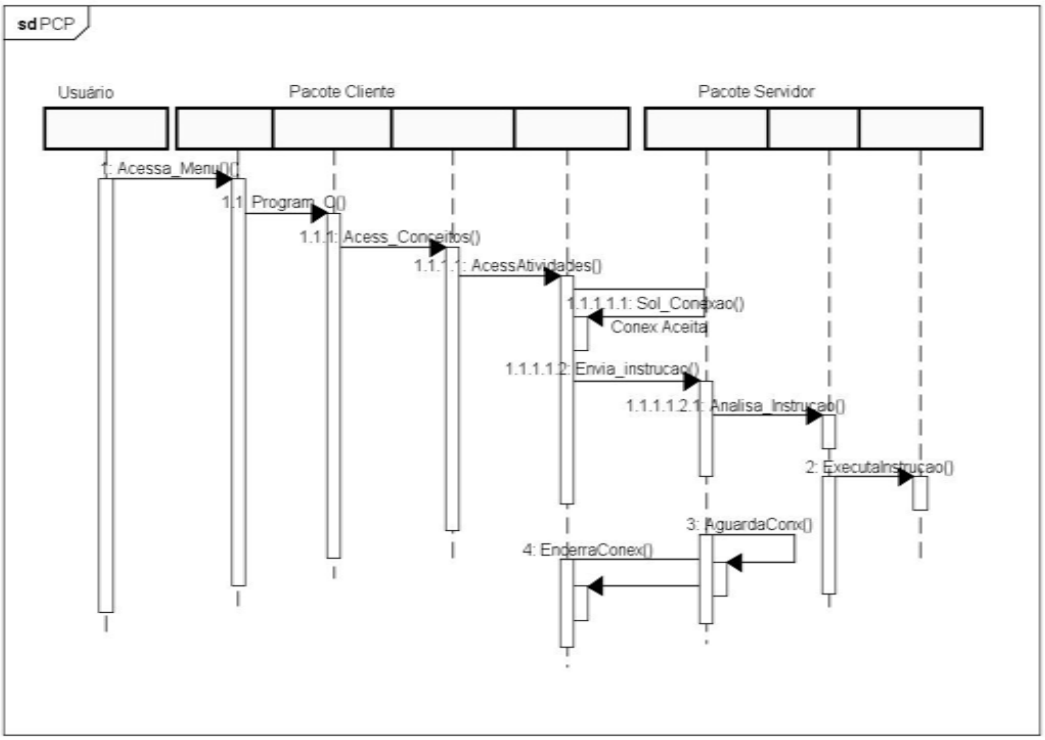


Figura 4.3 - Diagrama de Seguência (usuário implementado uma atividade)

A arquitetura do *framework* proposto é um ambiente que consiste em um conjunto de classes que visa manipular sensores, atuadores, variáveis, imagens, vídeos, áudio e documentos, sendo necessário ao usuário apenas navegar sobre as interfaces e implementar as instruções.

Esse ambiente é preparado para ser executado em Linux e *Windows*. A arquitetura do ambiente de aprendizagem descrita é representada por três pacotes principais que comprovam como a mesma pode ser executada. O primeiro pacote consiste de classes Java que implementam uma interface gráfica visando ministrar a teoria e a prática da disciplina de linguagem de programação C em um ambiente de computador.

Já o segundo |Pacote (comunicação) é responsável pela conexão entre o pacote de interface e o pacote PS. Além disso, é composto por classes Java e utiliza uma primitiva de transporte (*Socket*) do tipo *que* trafega na forma de *stream de caracteres* visando implementar os comandos enviados pelo primeiro pacote.

Além disso, utilizou-se o padrão *Ethernet* 802.11 para realizar a interconexão entre os dispositivos (notebook e o Robotino). O terceiro Pacote Servidor contém os métodos

reutilizáveis, ou seja, é um conjunto de funções que visa controlar os dispositivos do robô, (controladores, sensores, atuadores, manipuladores e etc). A Figura 4.4 ilustra a característica da arquitetura deste *framework*.

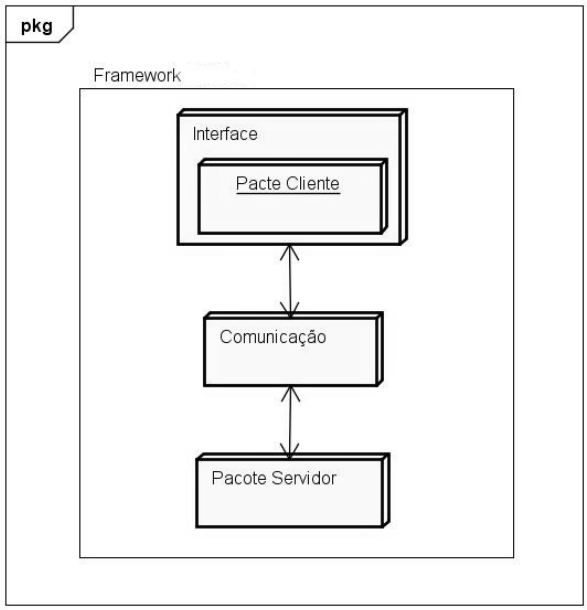


Figura 4.4 - Caracterização da arquitetura do *framework*

Após finalizar a modelagem da arquitetura do *framework*, o projeto direciona-se para a implementação do mesmo. Porém, esta abordagem não é apenas voltada para desenvolvimento de um programa de computador, ou seja, esta pesquisa se preocupa com toda a documentação associada e os dados de configuração necessários para fazer com que os programas operem corretamente em um ambiente de sala de aula.

Com isso, pode-se associar a qualidade desse *framework* como um conjunto de características que devem ser alcançadas para que o produto atenda às necessidades tanto do professor quanto do aluno. De certa forma, tem-se a questão das "exigências" pedagógicas, que incluem facilidades de uso deste *framework*. Assim, antes da implementação do *framework* proposto vamos esclarece a concepção de proposta.

4.5 Concepção da Proposta

O modo de interação da teoria (da linguagem de programação C) para a prática (via Robotino) é por meio da utilização do *framework* proposto. Ele é composto por um conjunto de pacotes visando o processo de ensino-aprendizagem de linguagem de

programação C, desenvolvido em Java e que pode ser adaptado a outras linguagens de programação. A Figura 4.5 demonstra o modelo do sistema composta por módulos e que estão relacionados entre si. Esse sistema é denominado nessa dissertação de mestrado de sistema FDRobô. Ele é composto pelo protótipo do *framework* (contendo os pacotes dos clientes, pacote de comunicação e pacote servidor que fica embarcado no Robotino), relação entre os módulos e metodologia de aprendizagem.

O módulo do ***framework*** (protótipo), ele contém o pacote interface, comunicação e servidor. Vale ressaltar que o dispositivo de saída é o próprio Robotino. Em suma, esses módulos são responsáveis pela funcionalidade do sistema e que se relacionam por meio de setas denominadas de relação. A ***relação*** acontece por meio da inserção de um conjunto de dados no pacote cliente, ambiente cooperativo, ambiente competitivo e pacote servidor. Esses dados são tratados no ambiente cooperativo para depois serem testados no ambiente competitivo por meio do dispositivo de saída (Robotino). O dispositivo de rede (comunicação) permite a conexão entre o *framework* (protótipo) e o dispositivo de saída (Robotino). Essa comunicação permite também que as atividades desenvolvidas tanto pelo aluno como pelo professor sejam executadas no Robotino. O módulo da ***metodologia de aprendizagem*** é composto pelo professor, aluno e os métodos de aprendizagens que são responsáveis pelas entradas dos dados no protótipo do *framework*.

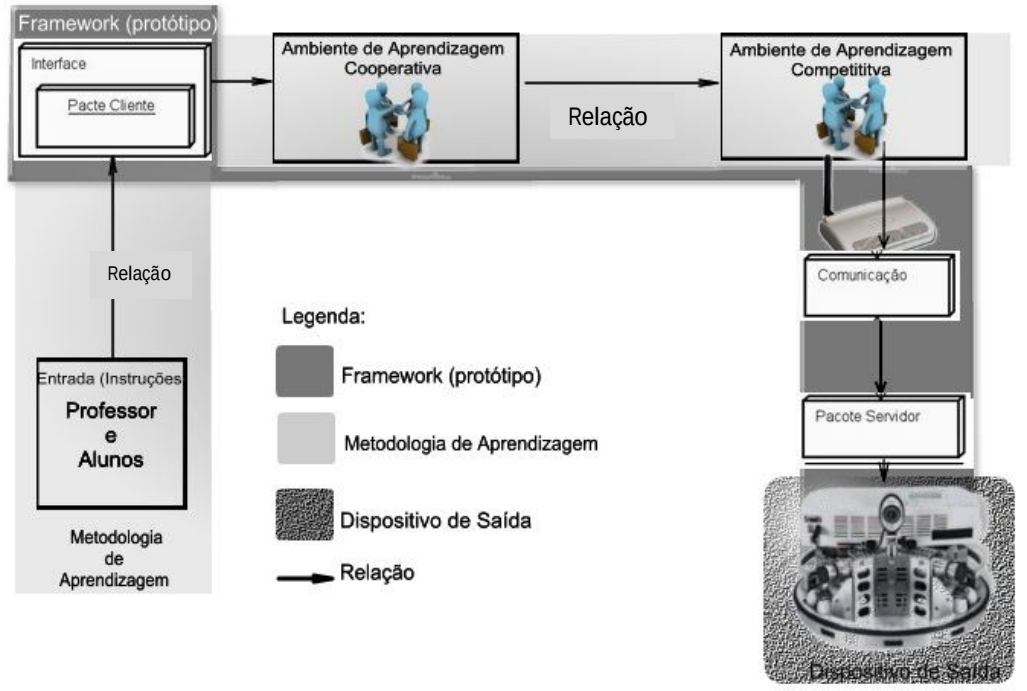


Figura 4.5 - Modelo do Sistema FDRobô. Fonte: Próprio Autor

4.6 Conclusão

A concepção da arquitetura foi responsável em definir o modelo do sistema proposto. Esta abordagem utiliza o modelo orientado a reúso. Assim, descreveu-se sobre especificação de requisitos, onde se definiu a linguagem de programação C como linguagem experimental para este projeto, a ementa utilizada pelo protótipo, o Robotino como o robô móvel para implementar os experimentos dos aluno e os requisitos de funcionalidade.

Na análise de componentes, realizou-se uma análise critica visando encontrar componentes para a construção do *framework* proposto. Assim definimos as regras necessárias para o funcionamento do *framework*. Já na fase de adaptações de requisitos, foi elaborada uma revisão dos requisitos das atividades “análise de componentes”. Os componentes módulo de interface, módulo de comunicação e módulo de controle foram adaptados para atender critérios pedagógicos.

O projeto de arquitetura do *framework* (protótipo) foi descrito por meio da apresentação de casos de uso, diagrama de classes e de sequência visando definir as atividades do futuro protótipo. E finalmente foi apresentada uma visão geral do modelo do sistema proposto visa facilitar o processo da busca de tecnologias que atendam os requisitos e permitam a construção dos protótipos. Este capítulo foi importante porque por meio dele foi possível modelar uma estrutura para o sistema proposto. Nos próximos capítulos vamos implementar esse modelo visando avaliar o sistema proposto.

Capítulo 5- Implementação do Modelo Proposto

Este capítulo apresenta um conjunto de atividades visando a implementação do *framework* proposto. A forma abordada para a construção do protótipo foi sobre a forma de componentes. Para isso, descrevemos os principais componentes como, Conteúdos, Atividade, Pesquisa e Multimídia visando o funcionamento do protótipo denominado de FDRobô.

Para elaboração da interface Homem-maquina, abordamos um estudo de caso baseado na usabilidade geral e pedagógica visando facilitar a condução do aplicativo pelo aluno/professor. A ideia é construir uma interface gráfica de fácil entendimento e condução.

Assim, as ferramentas, equipamentos e *softwares* escolhidos para auxiliar o desenvolvimento da arquitetura do aplicativo foram baseados no Estado da Arte. Ao final desse capítulo, serão demonstrados os testes de funcionalidades e de integração entre o protótipo e o Robotino.

5.1 Desenvolvimento do Protótipo

O *framework* desenvolvido nesta dissertação de mestrado tem como finalidade auxiliar o professor e o aluno no processo de ensino-aprendizagem de linguagem de programação C. O recurso disponível neste tipo de ambiente poderá possibilitar ao professor uma ferramenta de ajuda nas dificuldades dos alunos. Nas próximas seções estão descritos os componentes do *framework.* Vale ressaltar que as ferramentas que auxiliarão o desenvolvimento do protótipo estão descritas no Anexo III.

5.1.1 Protótipo

O protótipo é do tipo Cliente-Servidor, o aplicativo principal do cliente foi denominado nesta pesquisa de “FDRobô” (*Framework* Didático para manipular um robô),

pelas suas características de utilizar em sua base os conceitos de *Framework*, Didática e Robótica. Já o nome do aplicativo servidor foi denominado nesta pesquisa de "FDServe" (*Framework* Didático Para Servidor).

A Figura 5.1 descreve um diagrama de componentes visando demonstrar como serão implementadas as interfaces gráficas baseada nesse diagrama. No Pacote FDRobô contém o menu principal. Este possui 4 (quatro) componentes, o primeiro é o "Conteúdo", ele é responsável em conter os sub componentes "Aula_Robotino", "Aula_Conceito", "Aula_Varivel", "Aula_Operadores" e "Aula_Decisão", esses sub componentes contém os conteúdos teórico da disciplina de linguagem de programação C. Em seguida temos o componente "Atividade", é responsável pelo o escopo de implementação dos exercícios da disciplina linguagem de programação C. Em seguida os componentes de Pesquisa e de Multimídia, eles serão utilizados como um apoio didático (acesso a internet e vídeos). Nas próximas seções descrevemos as interfaces gráficas de cada componente por meio de um estudo de caso.

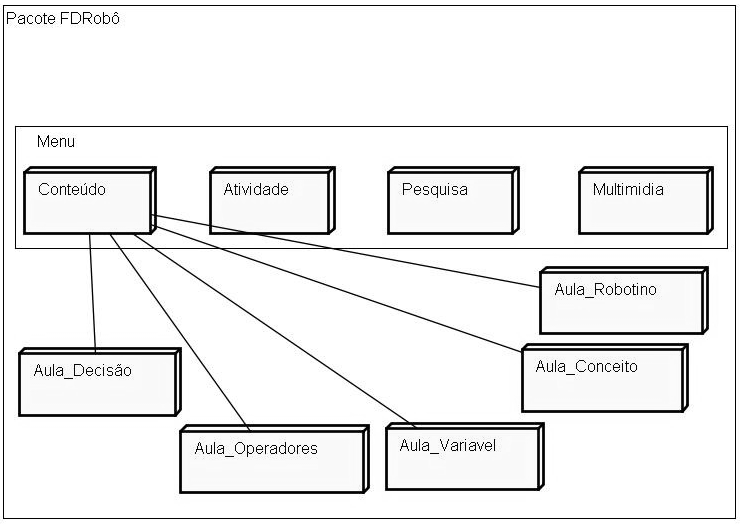


Figura 5.1 - Diagrama de componentes do FDRobô.

5.2 Estudo de Casos: Implementação do Protótipo

Foram definidos estudos de casos que são organizados de acordo com o diagrama da Figura 5.1. As telas do FDRobô foram baseadas na usabilidade pedagógica, ou seja, o que se buscou foi uma organização desses recursos para compor um cenário visando avaliar o *framework* em uma aula de linguagem de programação C.

Dessa forma, as interfaces gráficas implementadas em cada módulo são dispostas para o professor manipulá-las, não tendo como característica montar uma aula em tempo real de execução, preocupando-se apenas coma a interação com os alunos, pois os docentes não precisariam organizar os conteúdos em uma aula e sim escolher quais dos módulos do *framework* gostariam que fossem utilizados.

A Figura 5.2 demonstra o Menu principal do FDRobô, que é composto por quatro ícones principais e um console de ajuda. O Menu principal é o portão de entrada para os ícones Conteúdo, Atividade, Pesquisa e Multimídia. Ele se chama menu, pois oferece uma lista de opções, exatamente como o menu de um restaurante. E como a palavra "iniciar", ou seja, é o local onde você iniciará ou abrirá os itens.

Os testes foram implementados visando atender os requisitos da usabilidade pedagógica. Assim, ao passar o mouse sobre o ícone, um balão de mensagem sobre o ele descreverá no console as informações sobre o ícone ao usuário. Essas informações ajudam tanto o aluno como o professor na fácil condução do aplicativo. O ícone “Conteúdo” possibilita ao professor a inserção e exibição de slides relacionados à disciplina Linguagem de Programação C.

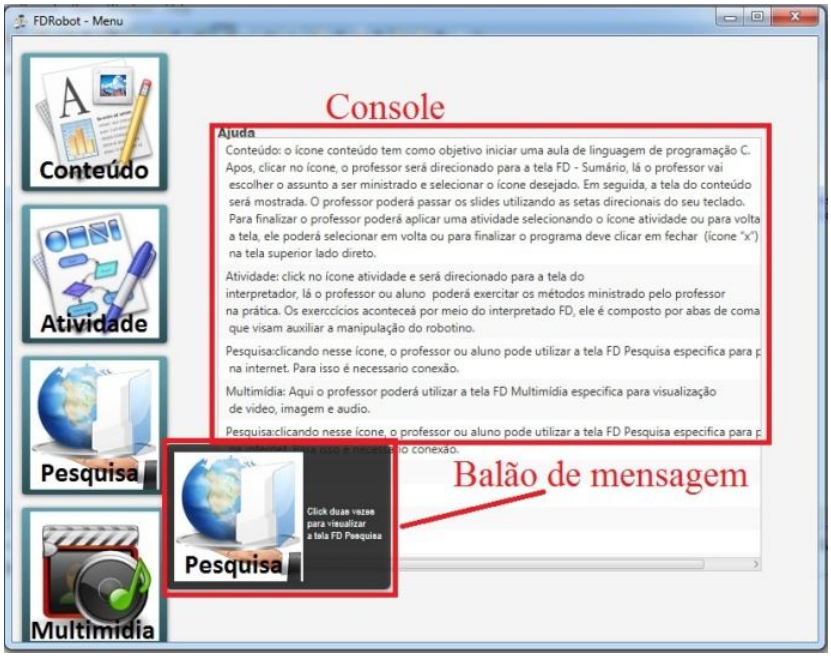


Figura 5.2 - Menu principal do *framework*

A interface da Figura 5.3, é uma sub tela do ícone “Conteúdo” da Figura 5.2 e foi elaborada para mostrar ao usuário o sumário do conteúdo de Linguagem de Programação C. Disponibiliza ao professor e ao aluno de forma geral os tópicos que podem ser

explorados como, “1-Conceitos”, “2-Métodos Variável”, “3-Métodos Operadores”, “4-Métodos Entrada Saída” e “5-Métodos Estrutura de Decisão”. Nesta interface o professor ou aluno podem escolher o conteúdo a ser estudado. Caso o usuário queira voltar ao Menu principal, deve-se clicar no ícone “Voltar”.

Figura 5.3 - Interface FDRobô - Prgramação C

O ícone “Pré-Aula Robotino” da Figura 5.3 tem como objetivo abrir a sub tela da Figura 5.4 e que contém uma apresentação de slides que possibilitará ao professor/aluno o ensino sobre o Robotino. Observe que nesta sub tela, tem-se os ícones inserir documento, inserir atividade, voltar ao menu principal e voltar ao sumário, eles são as opções do professor. O objetivo desta sub tela é ensinar os alunos os conceitos básicos de manipulação do Robotino. Além disso, o usuário usa as setas direcionais para passar o slide.

Figura 5.4 - Interface FDRobô - aula de Robotino

O ícone (1–Conceitos) da Figura 5.3 abre a sub tela “FDRobô – Conceitos” (Figura 5.5), ela contém os conceitos iniciais sobre linguagem de programação C. Nesta sub tela o

professor poderá utilizá-la para ministrar uma aula sobre os conceitos de linguagem de programação C. O professor pode também inserir documentos como apoio didático por meio do ícone “Inserir Documento”. Esta sub tela apresenta a primeira aula de linguagem de programação C utilizando o protótipo proposto. Após ministrar essa aula, o professor poderá demonstrar um exercício simples clicando no ícone “Inserir Atividade”. Além de ter outras opções como voltar ao sumário e ao menu principal.

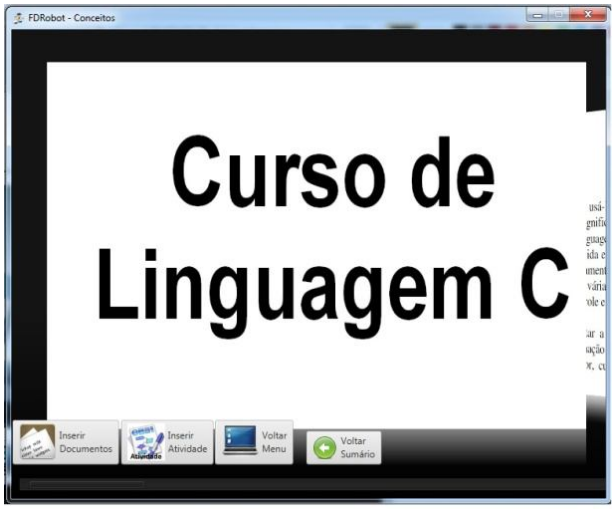


Figura 5.5 – Sub tela FDRobô – Conceitos

Já o ícone (2-Métodos Variável) da Figura 5.3, abre a sub tela “FDRobô – Variável” (Figura 5.6), que é responsável por apresentar o conteúdo sobre variável. Nesta sub tela o professor poderá utilizá-la para ministrar uma aula sobre os conceitos de variáveis, ou seja, pode ser uma opção para o ensino de variáveis da disciplina de linguagem de programação C. Utilizam-se as setas direcionais para passar os slides.

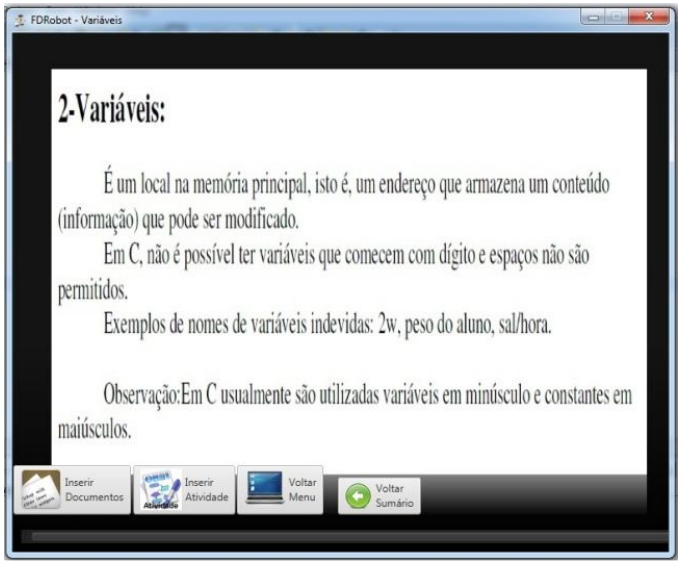


Figura 5.6 – Sub tela (FDRobô – Variável)

O ícone (3-Métodos Operadores) da Figura 5.3, abre a sub tela FDRobô – operadores (Figura 5.7). é responsável em apresentar os conceitos sobre operadores da linguagem de programação C. O objetivo desta sub tela é de oferecer ao professor a possibilidade de ministrar uma aula sobre operadores de Linguagem de Programação C. Utilizam-se as setas direcionais para passar os slides.

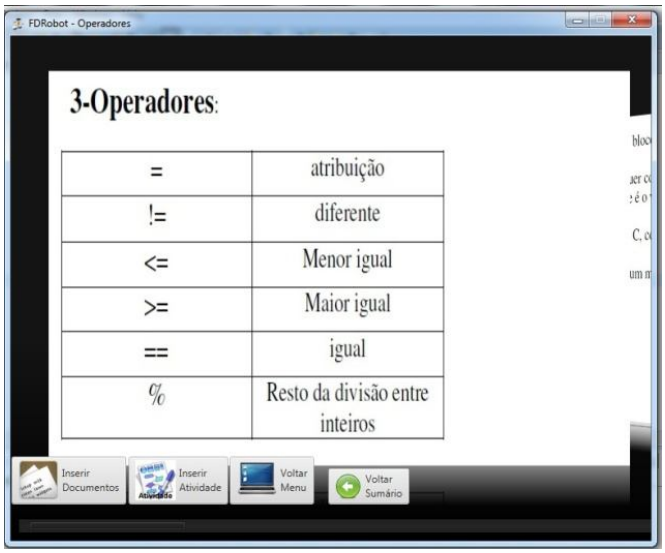


Figura 5.7 – Sub tela FDRobô – operadores

O ícone (4-Métodos Entrada e Saída) da Figura 5.3 abre a sub tela FDRobô – Entrada e Saída (Figura 5.8). O objetivo desta sub tela é auxiliar o professor a ministrar uma aula de comandos de impressão *print* da disciplina linguagem de programação C. Após isto o professor poderá demonstrar um exemplo prático por meio do ícone "inserir atividade".

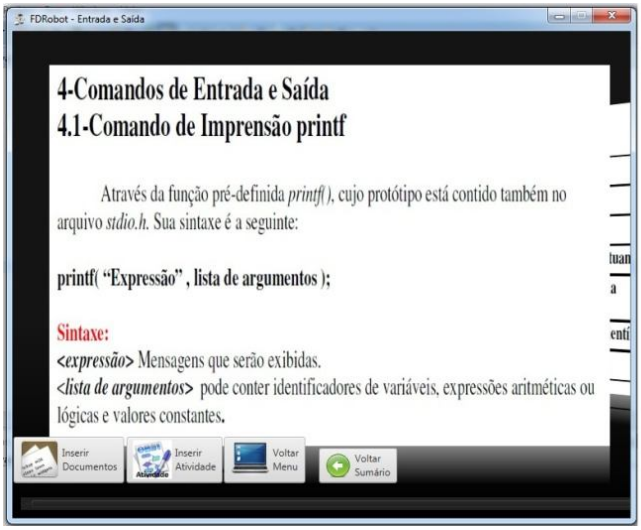


Figura 5.8 – Sub tela (FDRobô – Entrada e Saída)

O último ícone (5-Métodos Estrutura de Decisão) da Figura 5.3 abre a sub tela FDRobô – Estrutura de Decisão (Figura 5.9). O objetivo desta sub tela é de proporcionar ao professor um conjunto de slides apresentando os conteúdo de comando condicional if/else. De posse dessa ferramenta o professor tem a opção de decidir qual a melhor forma de inicializar o ensino de linguagem de programação C.

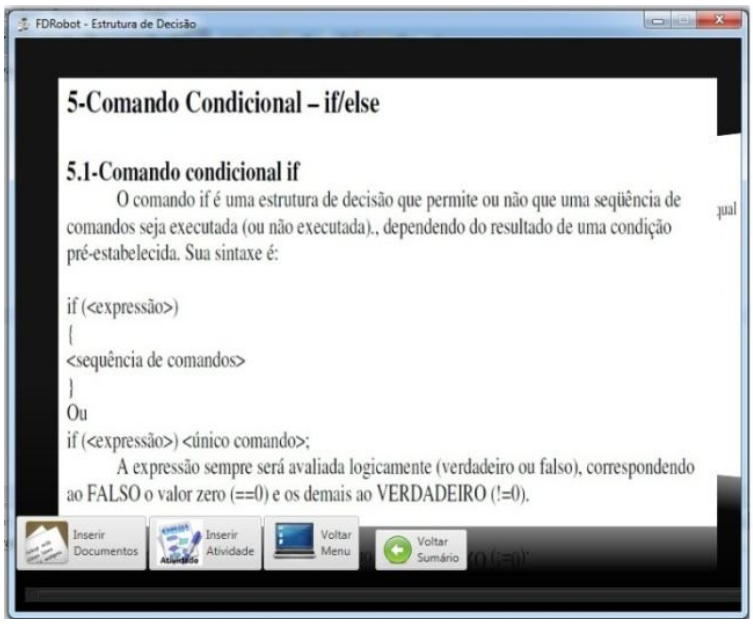


Figura 5.9 – Sub tela FDRobô – Estrutura de Decisão

A Figura 5.10 demonstra a tela denominada nesta pesquisa de FDRobô – Atividade. Essa tela tem como objetivo utilizar o Robotino na parte prática das atividades no laboratório. Para isso foi adicionado e desenvolvido os métodos de manipulação do Robotino. Primeiro adicionou-se a paleta “Operadores”, ela contém os operadores aritméticos (+, -, *, /) onde pode ser utilizado nas instruções de Linguagem de Programação C.

A segunda paleta possui o “Controle”, contendo os métodos “printf” responsável pela saída das instruções no Robotino. Na paleta “Funções do Robotino” foram desenvolvidas as técnicas específicas de medidas angulares (ver apêndice C) visando o deslocamento do Robotino no plano. Na paleta “Variáveis Robotino” têm-se as variáveis particulares do Robotino visando a sua utilização nas instruções de Linguagem de Programação C.

A paleta C-Básico é o escopo de programação da Linguagem de Programação C e conterá as instruções e/ou expressões, que definem um algoritmo executável ou parte dele. Além disso, temos o ícone de “Limpar” (responsável em excluir tudo o que foi escrito no

escopo, botão “Executar” (responsável pela implementação das instruções no Robotino) e botão “Conexão Robotino” (responsável em conectar o Robotino ao aplicativo).

.

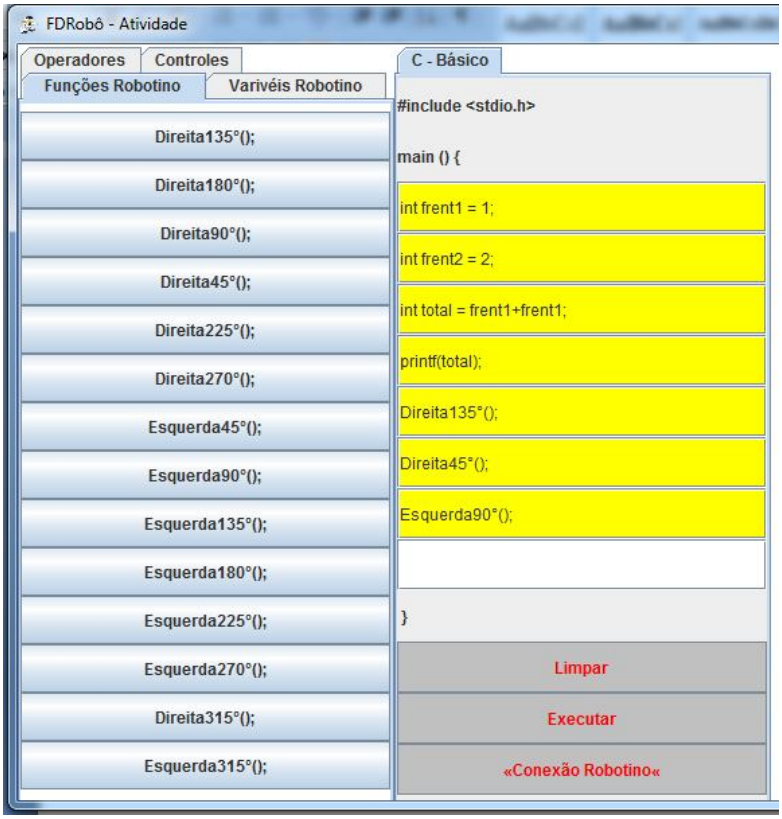


Figura 5.10 - Interface FDRobô – Atividade

A Figura 5.11 contém a tela de interface gráfica FDRobô – Pesquisa, que é um mecanismo de acesso à Internet que visa apoiar o professor e/ou o aluno nas pesquisas via *web*. Vale ressaltar, que esta interface requer conexão com a internet. Assim, o professor não precisa abrir outro programa de computador para fazer uma pesquisa, pois isso pode desviar o foco dos alunos. A ideia é proporcionar um aplicativo que facilite o processe de ensino-aprendizagem. Na Figura 5.12 temos a tela denominada neste trabalho de “FDRobô – Multimídia”, nela temos três ícones utilizados para abrir documentos (Doc, PDF, TXT e outros), vídeo (JPEG, MP4, FLV e outros) e áudio (WAVE, MP3, MP4 e outros).

Figura 5.11 - Interface FDRobô - Pesquisa

Figura 5.12 - Interface FDRobô - Multimídia

Para o desenvolvimento do protótipo FDServe, utilizou-se um *socket* para se comunicar com FDRobô. Os processos remotos usando *sockets* utilizam a rede (TCP/IP) por meio de *Datagrama* (unidade básica de dados). Assim, a Figura 5.13 descreve o Notebook com o protótipo FDRobô, utilizando um canal de comunicação remoto (*sockets)* com "FDServe" embarcado no Robotino, ou seja, o "FDRobô" envia os comandos por meio da comunicação para o" FDServe".

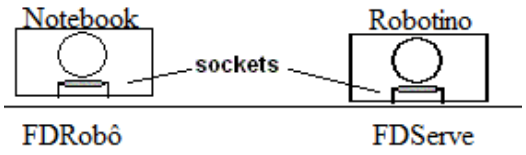


Figura 5.13- Conexão *Sockets*

5.3 Teste de Integração do FDRobô, FDServe e Robotino

Para integrar o FDServe ao sistema do Robotino utilizamos o Eclipse por meio dos seguinte passos: Passo 1 – Utilizem o RSE (*Remote System Explorer*) do Eclipse IDE. O RSE é uma ferramenta recente do Eclipse que possibilita o acesso e o gerenciamento remoto do Robotino de forma fácil e tudo pela interface do Eclipse. Existe também o método utilizado pelo Robotino, mas é bastante complicado porque utilizamos linhas de comandos para ter acesso aos arquivos.

Passo 2 – Inicie abrindo a perspectiva do RSE em "Window->Open Perspective->Other->RSE Explorer" selecione uma conexões disponíveis no Eclipse. Crie uma conexão remota via Secure Shell (SSH - é um protocolo que permite o acesso virtual ao servidor como se você estivesse em um terminal [53]) com o Robotino. Esta configuração exige que a maquina esteja conectada a rede do Robotino (ligamos o robô e verificamos qual o IP da rede do Robotino). Passo 3 - Clique no botão direito da janela à esquerda que contém as conexões disponíveis e selecione "*New->Connection*". Em seguida selecione "SSH *Only*" e clique em "*Next*". Digite o IP do Robotino no campo "*Host name*" e define um nome para a conexão "*Connection name*". Clique em "*Finish*". Em seguida outra conexão com o nome definido irá apareceu à esquerda. Passo 4 - Clique com o botão direito nela e selecione "*Connect*", digite o nome de usuário (Robotino) e senha (Robotino). Se não receber nenhuma mensagem de erro, o acesso remoto foi configurado com sucesso. Passo 5 – Copie a pasta do FDServe do seu computador para o Robotino. Passo 6 – Agora abra o *Launch* do Terminal e clique com o mouse direito sobre o SSH Terminal. Passo 7 – No terminal digite os comandos -> Cd examples-> Cd FDserver-> ./ FDserver 127.0.0.1 e digite ENTER. Simultaneamente verifique se o Robotino acendeu o display e mostrou o IP. Esses passos foram feitos várias vezes, justamente para chegar a uma conexão estabelecida com sucesso entre o FDserve e o Robotino.

Com o FDServe executando, vamos conectar o FDRobô ao FDserve. Para isso, seguimos os seguintes passos: Passo 1 – Abra o aplicativo ->FDRobô – Menu -> Ícone Atividade. Passo 2 – Execute o ícone "Conexão Robotino" e confirme o "OK" da mensagem (Figura 5.14).

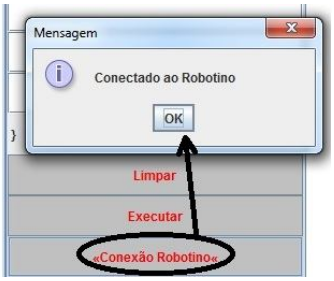


Figura 5.14 - Conexão entre FDRobô e Robotino

Passo 3 – Além dessa confirmação verifique se no terminal o FDServe descreveu a seguinte mensagem no Terminal IDE "Conectado FDRobô 127.0.0.1", se tudo foi configurado corretamente, a conexão será estabelecida com sucesso entre o FDRobô, FDServe e o Robotino. Nas próximas seções vamos testar algumas atividades de Linguagem de Programação C.

5.4 Funcionalidade do Protótipo FDRob, FDServe e Robotino

Os primeiros testes buscam verificar as funcionalidades do protótipo proposto e foi realizado pelo próprio desenvolvedor. O diagrama de atividade (Figura 5.15) descreve como os testes aconteceram por meio de inúmeras tentativas, utilizando exercícios de linguagem de programação C (Quadro 5.1). Assim, inicia-se o aplicativo, seleciona-se o ícone Conteúdo, escolher a Aula_Variavel, depois seleciona-se o ícone atividade, soluciona-se os exercícios e verifica se o exercício foi executado com sucesso ou sem sucesso.

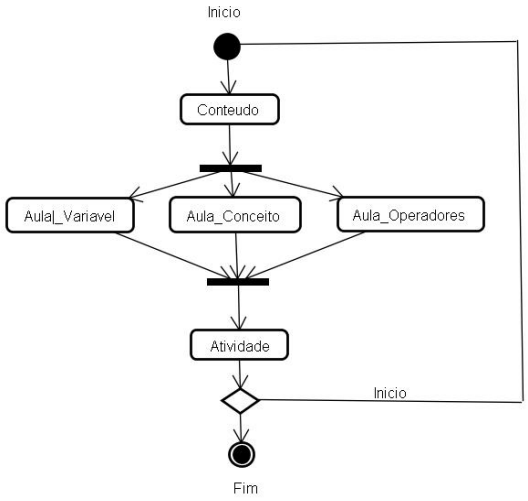


Figura 5.15 - Diagrama de atividade

O Quadro 5.1 descreve que o *framework* está funcionando, pois o exercício de nº 1 (um) teve 10 (dez) experimentos e comprovou que 92% dos experimentos tiveram sucesso e 8% de execuções sem sucesso devido a perda de conexão wireless. No exercício de nº 2 (dois) observou-se que de 10 (dez) experimentos 96% das instruções foram executados com sucesso, enquanto 4% não foram executados com sucesso devido a fatores como, erro de memória do computador e no último experimento tivemos 10 (dez) experimentos, mas apenas 93% executaram com sucesso e 7% não tivemos sucesso por causa de comandos errados. Para saber os comandos utilizados nesses exercícios ver apêndice C no final dessa dissertação.

Quadro 5.1 - Resultados dos experimentos

Nº	**Exercício**	**Número de tentativas**	**Execução com sucesso**	**Execução sem sucesso**
01	1.º Faça a variável "frente1" reservada do Robotino ser uma variável do tipo inteiro e mover o robô em duas (2) posições no plano; c) A variável deve mover o robô 5 (cinco) posições no plano.	10	92%	8%
02	2.º Faça a variável "frente" reservada do Robotino ser uma variável do tipo inteiro e mover o robô em duas (2) posições frente e duas posição esquerda;	10	96%	4%
03	3.º Faça o Robotino se deslocar no plano utilizando o operador adição (+). A variável deve mover o robô duas vezes a sua posição atual no plano;	10	93%	7%

5.5 Conclusão

Neste capítulo foi apresentado um conjunto de atividades que resultou no desenvolvimento do *framework* proposto. Em suma, foram apresentadas as ferramentas, equipamentos e *softwares* escolhidos para desenvolver o arcabouço do aplicativo. Em

seguida foram construídos o protótipo denominado FDRobô e FDServe para compor o sistema do *framework.*

As interfaces Homem-máquina foram baseadas na usabilidade geral e pedagógica, pois os objetivos das interfaces são de oferecer um aplicativo de fácil condução e aprendizagem tanto para o professor quanto para o aluno. Em seguida foi desenvolvido um conjunto de métodos para manipular o Robotino e ao final foram descritos os testes de integração entre o protótipo e o Robotino visando verificar sua eficiência e eficácia.

Os primeiros testes das funcionalidades do protótipo demonstraram que mais de 90% das implementações foram concluídas com sucesso e por meio desses experimentos podemos afirmar que o *framework* está funcionando. Assim, o próximo passo será de avaliar o *framework* proposto junto aos alunos.

Capítulo 6- Avaliação e Resultados

Este capítulo avalia o *framework* proposto nos capítulos anteriores. Será apresentado um conjunto de procedimentos para avaliar o sistema proposto como, a descrição da metodologia adotada, perfil dos participantes, descrição dos ambientes de aprendizagem cooperativo e competitivo, avaliação do protótipo nos ambientes de aprendizagem e as avaliações de usabilidade geral e pedagógico no sistema FDRobô.

Para isso, descrevemos a metodologia utilizada na avaliação do sistema. As formas adotadas para as realizações das avaliações foram por meio dos questionários de usabilidade geral e pedagógica (Anexo II e III) utilizando uma série de perguntas junto aos alunos.

As análises têm como principal característica apresentar os resultados para serem comentados com os resultados apresentados pelos trabalhos de referências. O sistema foi avaliado em relação ao desempenho da turma, à sua operabilidade de utilidade e ao grau de satisfação do grupo de estudantes em relação ao mesmo.

6.1 Descrição Detalhada dos Procedimentos Adotados na Metodologia Proposta

Os testes foram realizados com a finalidade de garantir a Usabilidade Geral e Pedagógica do sistema FDRobô. Dessa forma, identificamos os perfis dos alunos (sub seção 6.1.1), orientamos sobre os ambientes de testes, explicando o propósito das avaliações e como funciona o sistema FDRobô. Além disso, será descrito o ambiente de aprendizagem cooperativo (sub seção 6.1.3) e o ambiente de aprendizagem competitivo (sub seção 6.1.4)

O professor iniciará o curso de linguagem de programação C utilizando o aplicativo FDRobô composto pelos conceitos de linguagem de programação C. Em seguida o professor utilizará dois ambientes para avaliar o aplicativo FDRobô tanto no ambiente de aprendizagem cooperativa (Seção 6.2) como no ambiente de aprendizagem competitiva (Seção 6.3). Vale ressaltar, que as tarefas do Quadro 6.3 serão utilizadas como

experimentos em ambos ambientes. Os resultados dessas avaliações serão comentados com outros trabalhos (Seção 6.4). Finalmente, o sistema FDRobô será avaliado por meio dos princípios estabelecidos pela Norma ISO 9241- Parte 10 – Princípios de Diálogo, baseado nas dez Heurísticas de usabilidade, propostos por Nielsen [35] (Anexo III), visando medir a usabilidade geral e para os critérios de usabilidade pedagógica utilizou-se o método de Nokelainen [37] (Anexo II). Logo após, finaliza-se a avaliação.

6.1.1 Perfil dos Participantes e Deveres do Professor

Segundo os Princípios de Diálogo [53], o planejamento para o teste de usabilidade geral inclui informações sobre o produto, aplicabilidade do produto e a caracterização dos usuários. Para a seleção do pessoal foi solicitada autorização prévia por meio de um plano de avaliação para executar a pesquisa com seres humanos de acordo com os requisitos da Resolução CNS 466/12 da Comissão de Ética da Universidade Federal do Amazonas (UFAM). Esta recebeu aprovação por meio do CAAE 49715315.9.0000.5020 e número do parecer: 1.320.058 para sua realização [6].

Os experimentos contaram com 15 participantes, com idade entre de 18 a 25 anos, nível médio completo, são calouros de uma faculdade particular em Manaus, mais de um ano de conhecimentos básicos de informática e nenhum conhecimento em linguagens de programação. O professor é responsável em observar os participantes durante a realização das avaliações. Dessa forma, o professor não pode ajudar o participante na realização dos exercícios. Ele somente orienta se surgir uma questão acerca do procedimento das atividades.

6.1.2 Ambiente de Suporte ao Ensino

As Figuras 6.1 e 6.2 demonstram o ambiente de testes, ele foi equipado com uma câmera para o registro do evento, sendo que ela está posicionada a frente do participante. O sistema FDRobô está pronto para ser utilizado, ou seja, o aplicativo FDRobô e FDServe e o Robotino estavam prontos para realizar todas as funcionalidades no computador. Mesas e cadeiras foram organizadas de acordo com as necessidades das equipes. Utilizamos um *data show* para projetar o aplicativo FDRobô e delimitamos uma área para as competições entre as equipes.

Figura 6.1-Ambiente de testes

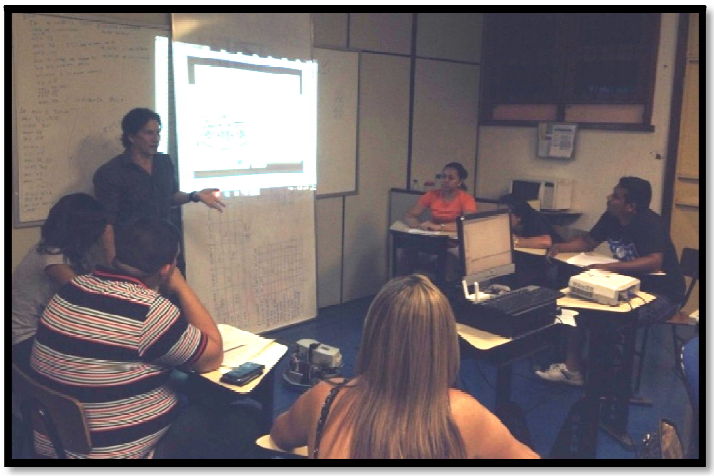


Figura 6.2 - Orientações do teste

6.1.3 Ambiente de Aprendizagem Cooperativo

A aprendizagem cooperativa é um método educacional onde os alunos ajudam e confiam uns nos outros para atingir um objetivo definido. Os 15 (quinze) alunos participantes foram divididos em três equipes (A, B e C). A Figura 6.3 demonstra um diagrama de atividades integrando o método de aprendizagem cooperativa ao professor, aluno, aplicativo FDRobô e Robotino. Nesse ambiente as equipes são formadas no início do curso e o professor é responsável em mediar a escolha dos líderes de equipe aleatoriamente para depois os lideres selecionarem o resto dos membros de sua equipe por meio de um sistema de "rodízio". Em seguida o professor implementa o aplicativo FDRobô e ministra uma aula de linguagem de programação C e ensina como solucionar

problemas utilizando o aplicativo no Robotino. Simultaneamente as equipes assimilam às aulas. Logo após, o professor apresenta uma lista de atividades (Quadro 6.1- Seção 6.2) para as equipes solucionarem. As equipes executam o FDRobô e buscam a melhor solução baseado no tempo. As equipes entregam as soluções e finalizam as atividades.

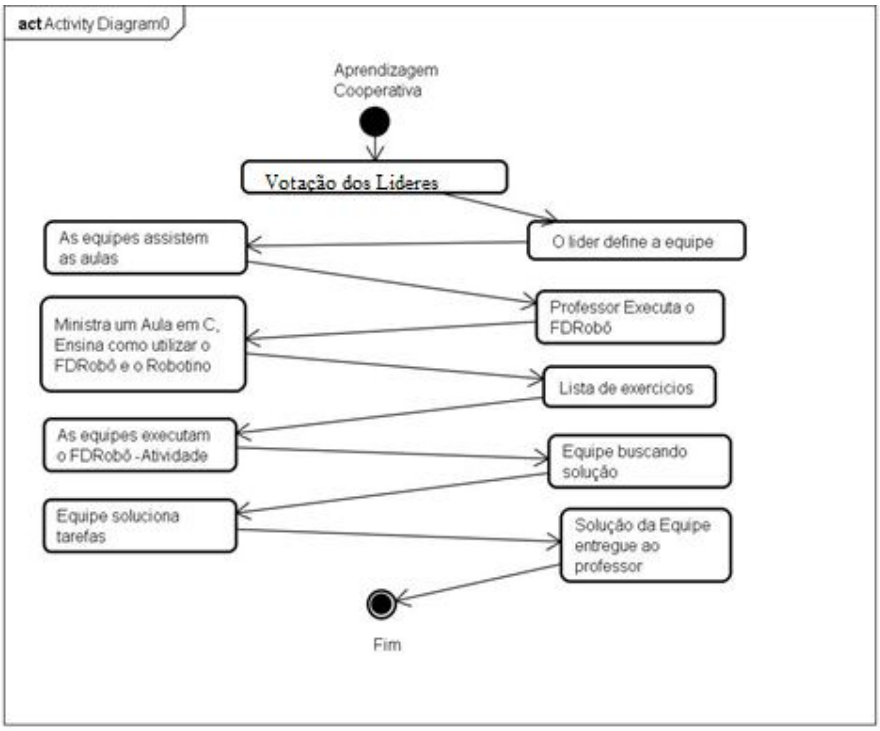


Figura 6.3 – Ambiente de Aprendizagem cooperativa, Fonte: Próprio Autor.

6.1.4 Ambiente de Aprendizagem Competitivo

O objetivo dessa competição é assumir uma forma lúdica no processo de ensino-aprendizagem. Uma competição saudável traz vários benefícios como a socialização entre os alunos, eles aprendem a lidar com diversas situações, desenvolve nos alunos valores como respeito, cooperação e solidariedade. Vale ressaltar que não existe um tempo definido para as soluções dos problemas apresentados pelo professor. Quem vai definir o melhor tempo é o empenho de cada equipe. Além disso, a equipe deve analisar todos os itens (energia do Robotino, cooperação entre os membros da equipe) necessários para ter um bom desempenho. Em suma, o segundo cenário competitivo (Figura 6.4) descreve a equipe ‘A’ e ‘B’ aplicando as soluções encontradas no ambiente cooperativo no FDRobô por meio do Robotino. O professor identifica o melhor tempo por meio do cronômetro e apresenta o melhor resultado à equipe.

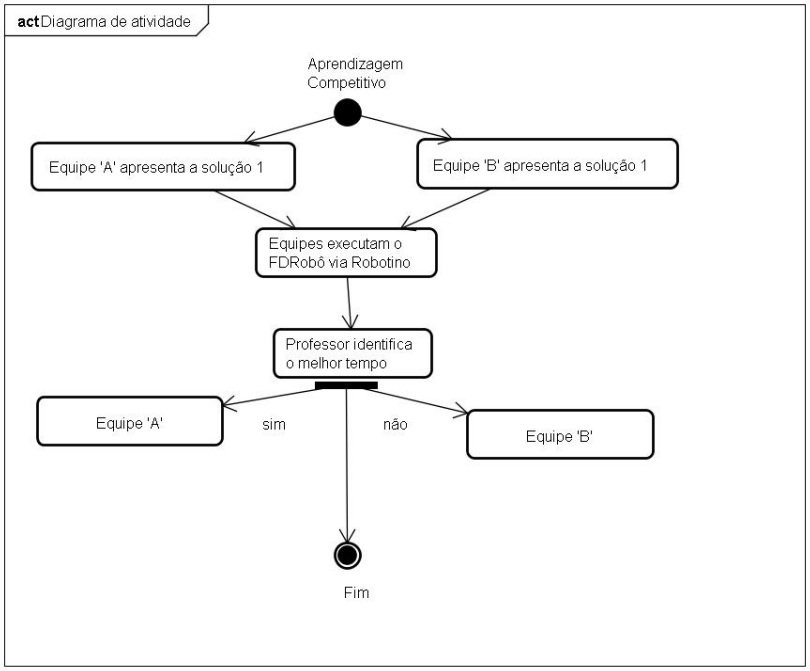


Figura 6.4 – Ambiente de Aprendizagem Competitiva. Fonte: Próprio Autor.

Os resultados são entregues de forma independente para a equipe e a pontuação é calculada ou inferida pelo professor, como médias individuais. O professor determina a pontuação da equipe por meio da *Média do tempo gasto para completar a tarefa* – calculada pela somatória do tempo gasto por todos os participantes (das equipes) que completaram a tarefa. Para calcular o tempo utilizamos a seguinte formula:

Desvio Padrão (S) – O desvio padrão é uma medida de variabilidade dos dados, ou seja, em que grau os dados se diferem uns dos outros. Para calcular esses fatores utilizou-se a seguinte fórmula:

$$S = \sqrt{\frac{\sum x^2 - \frac{\left(\sum x\right)^2}{n}}{n-1}}$$

Onde: $\sum x^2$ => soma do quadrado de cada um dos tempos coletados

$\sum x$ => soma de todos os tempos coletados

n => total de participantes

Neste tipo de atividade, o professor pode selecionar um dos membros da equipe para apresentar a solução para todos os alunos.

A Figura 6.5 descreve o ambiente de competição entre equipes utilizando o Robotino. Nesse ambiente definimos o máximo de deslocamento do Robotino por meios dos métodos específicos desenvolvido para o Robotino.

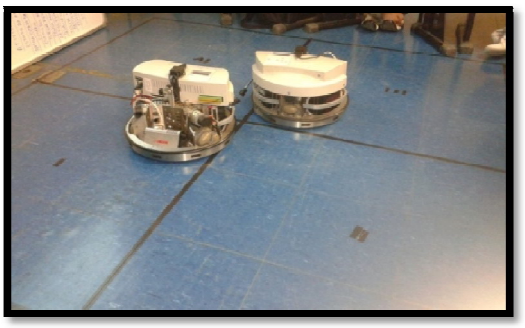


Figura 6.5 – Competição entre equipes

6.2 Avaliação do FDRobô com Aprendizagem Cooperativa

O professor elabora uma lista de atividades visando à construção dos conceitos de linguagem de programação C nos alunos. O Quadro 6.1 descreve os métodos envolvidos nas tarefas visando que as equipes solucionem todas as tarefas.

Quadro 6.1 - Lista de Atividades

Método aplicado	**Nº**	**Tarefas**	**Objetivo da atividade**
Variáveis e função printf	T1	1.º Faça a variável "frente1" reservada do Robotino ser uma variável do tipo inteiro e mover o robô em duas (2) posições no plano; c) A variável deve mover o robô 5 (cinco) posições no plano.	Abordar os conceitos de variáveis, tipos de variáveis e função printf.
	T2	2.º Faça a variável "frente" reservada do Robotino ser uma variável do tipo inteiro e mover o robô em duas (2) posições para frente e duas posição a esquerda;	
Variáveis e Operadores e função printf.	T3	3.º Faça o Robotino se deslocar no plano utilizando o operador adição (+). A variável deve mover o robô duas vezes a sua posição atual no plano;	Praticar os conceitos de variáveis e operadores.
	T4	4.º Faça o Robotino se deslocar no plano utilizando o operador de multiplicação (*): a) Defina uma variável reservada do Robotino e multiplique ela por 2; b) A variável deve mover o robô no plano;	
Variáveis, Operadores, Expressões Estrutura de decisão.	T5	5º Utilize o If e um operador relacional para determinar a tomada de decisão de uma variável e depois mostre o resultado.	Praticar os conceitos de variáveis e operadores e estrutura de decisão

As Figuras 6.6, 6.7 e 6.8 descrevem os dados das atividades implementadas pelas equipes A, B e C. Primeiro apresentamos a Figura 6.6, que descreve os resultado da equipe A. As colunas com os números de alunos (AlunoA1, AlunoA2, AlunoA3, AlunoA4 e AlunoA5) descrevem o tempo. As linhas (Tarefa1, Tarefa2, Tarefa3, Tarefa4 e Tarefa5) descrevem o tempo executado por cada aluno nas suas respectivas tarefas. Por exemplo, o "Aluno1" executou na sua "Tarefa1", no tempo de um minuto e quarenta e cinco segundos (00:01:45). Já o AlunoA2 implementou a Tarefa2 em um minuto e quarenta e oito segundos (00:01:48).

	AlunoA1	AlunoA2	AlunoA3	AlunoA4	AlunoA5
Tarefa1	00:01:45	00:00:00	00:00:00	00:00:00	00:00:00
Tarefa2	00:00:00	00:01:48	00:00:00	00:00:00	00:00:00
Tarefa3	00:00:00	00:00:00	00:01:50	00:00:00	00:00:00
Tarefa4	00:00:00	00:00:00	00:00:00	00:01:59	00:00:00
Tarefa5	00:00:00	00:00:00	00:00:00	00:00:00	00:02:05

Figura 6.6 - Dados quantitativos da equipe A

A Figura 6.7 da equipe B descreve um Quadro contendo os alunos, o tempo e as tarefas. As colunas contêm o número de alunos (AlunoB1, AlunoB2, AlunoB3, AlunoB4 e AlunoB5). As linhas (Tarefa1, Tarefa2, Tarefa3, Tarefa4 e Tarefa5) descrevem o tempo executado por cada aluno nas suas respectivas tarefas. Por exemplo, o "AlunoB5" executou na sua "Tarefa5" o tempo de dois minutos e sete segundos (00:02:07).

	AlunoB1	AlunoB2	AlunoB3	AlunoB4	AlunoB5
Tarefa1	00:01:46	00:00:00	00:00:00	00:00:00	00:00:00
Tarefa2	00:00:00	00:01:47	00:00:00	00:00:00	00:00:00
Tarefa3	00:00:00	00:00:00	00:01:51	00:00:00	00:00:00
Tarefa4	00:00:00	00:00:00	00:00:00	00:01:58	00:00:00
Tarefa5	00:00:00	00:00:00	00:00:00	00:00:00	00:02:07

Figura 6.7 - Dados quantitativos da equipe B

Para a equipe C, a Figura 6.8 descreve o tempo associado às tarefas dos alunos. O numero de alunos (AlunoC1, AlunoC2, AlunoC3, AlunoC4 e AlunoC5) descreve o tempo de cada aluno As linhas (Tarefa1, Tarefa2, Tarefa3, Tarefa4 e Tarefa5) descrevem o tempo executado por cada aluno nas suas respectivas tarefas. Por exemplo, o "AlunoC3" executou na sua "Tarefa3" o tempo de um minuto e cinquenta e um segundo (00:01:51).

	AlunoC1	AlunoC2	AlunoC3	AlunoC4	AlunoC5
Tarefa1	00:01:47	00:00:00	00:00:00	00:00:00	00:00:00
Tarefa2	00:00:00	00:01:46	00:00:00	00:00:00	00:00:00
Tarefa3	00:00:00	00:00:00	00:01:53	00:00:00	00:00:00
Tarefa4	00:00:00	00:00:00	00:00:00	00:01:59	00:00:00
Tarefa5	00:00:00	00:00:00	00:00:00	00:00:00	00:02:03

Figura 6.8 - Dados quantitativos da equipe C

6.3 Avaliação do FDRobô com Aprendizagem Competitiva

Os resultados encontrados para a Equipe (A):

Os tempos gastos para completar todas as tarefas estão descritos no Quadro 6.2. Assim, a média do tempo desse quadro foi 1.614s. Observando ainda esse quadro encontramos a diferença do tempo mais alto (2,05) e o tempo mais baixo (1,45) gasto para se completar uma tarefa.

Quadro 6.2 - Média de tempo das tarefas Equipe (A)

Equipe (A)	x = tempo necessário para realizar a tarefa (em minutos)
AlunoA1	1,45
AlunoA2	1,48
AlunoA3	1,50
AlunoA4	1.59
AlunoA5	2,05
n = 5	$\sum x$ = 8,07

$\sum x/n$ => 8,07/5 => 1.614 s.

Baseando-se nos dado do Quadro 6.2 calculou-se o Desvio Padrão (S): $\sum x$ e de $\sum x^2$, apresentados no Quadro 6.3:

$$S = \sqrt{\frac{\sum x^2 - \frac{\left(\sum x\right)^2}{n}}{n-1}}$$

Quadro 6.3 – Cálculo do Desvio Padrão das tarefas da equipe (A)

Equipe (A)	x = tempo necessário para realizar a tarefa (em minutos)	x^2
AlunoA1	1,45	2.10
AlunoA2	1,48	2.19
AlunoA3	1,50	2.25
AlunoA4	1.59	2.52
AlunoA5	2,05	4.20
	$\sum x$=8,07	$\sum x^2$= 13,26

$$S = \sqrt{13{,}26 - \frac{(8{,}07)^2}{5}} \Big/ \frac{}{5-1} => \sqrt{\frac{13{,}26 - 13{,}024}{4}} => \sqrt{\frac{0{,}236}{4}} = 0{,}05 \; milésimo\ de\ segundos$$

Concluiu-se que o desvio padrão presente na média aritmética da equipe (A) foi de 1.614 ± 0,05, ou seja, para este caso o erro pode ser ± 0,05 sobre a média.

Os dados encontrados para a Equipe B:

Os tempos gastos para completar todas as tarefas estão descritos no Quadro 6.4. Assim, a média do tempo de todas as tarefas foi 1.618s, observando o quadro encontramos a diferença do tempo mais alto (2,07) e o tempo mais baixo (1,46) gasto para se completar uma tarefa.

Quadro 6.4 - Média de tempo das tarefas da Equipe (B)

Equipe (A)	x = tempo necessário para realizar a tarefa (em minutos)
AlunoB1	1,46
AlunoB2	1,47
AlunoB3	1,51
AlunoB4	1.58
AlunoB5	2,07
n = 5	∑x = 8,09

∑x/n => 8,09/5 => 1.618s

Baseando-se nos dados do Quadro 6.4, logo temos o cálculo de $\sum x$ e de $\sum x^2$, apresentados no Quadro 6.5:

$$S = \sqrt{\frac{\sum x^2 - \frac{\left(\sum x\right)^2}{n}}{n-1}}$$

Quadro 6.5 – Cálculo do Desvio Padrão das tarefas da Equipe (B)

Equipe (B)	x = tempo necessário para realizar a tarefa (em minutos)	x^2
AlunoB1	1,46	2.13
AlunoB2	1,47	2.16
AlunoB3	1,51	2.28
AlunoB4	1.58	2.49
AlunoB5	2,07	4.28
n = 5	$\sum x$=8,09	$\sum x^2$= 13,34

$$S=\sqrt{13{,}34-\frac{(8{,}09)^2}{5}}{5-1} => \sqrt{\frac{13{,}34-13{,}089}{4}} => \sqrt{\frac{0{,}250}{4}} = 0{,}06\ milésimo\ de\ segundos$$

Para a equipe (B), podemos ver a utilização do desvio padrão na apresentação da média aritmética, informando o quão “confiável” o valor de 1.618 ± 0,06.

Os dados encontrados para a Equipe (C):

Os tempos gastos para completar todas as tarefas estão descritos no Quadro 6.6. Assim, a média do tempo de todas as tarefas foi 1.628s. Observando o quadro encontramos a diferença do tempo mais alto (2,05) e o tempo mais baixo (1,49) gasto para se completar uma tarefa.

Quadro 6.6 - Média de tempo das tarefas da Equipe (C)

Equipe (C)	x = tempo necessário para realizar a tarefa (em minutos)
AlunoC1	1,49
AlunoC2	1,46
AlunoC3	1,55
AlunoC4	1.59
AlunoC5	2,05
n = 5	∑x = 8,14

∑x/n => 8,14/5 => 1.628s

Assim, a média do tempo gasto para completar a tarefa foi 1.628s. Baseando-se nos dados do Quadro 6.6, logo temos o cálculo de $\sum x$ e de $\sum x^2$, apresentados no Quadro 6.7:

Quadro 6.7 – Cálculo do Desvio Padrão das tarefas da Equipe (C)

Equipe (C)	x = tempo necessário para realizar a tarefa (em minutos)	x^2
AlunoC1	1,49	2.10
AlunoC2	1,46	2.19
AlunoC3	1,55	2.25
AlunoC4	1.59	2.52
AlunoC5	2,05	4.20
	$\sum x$=8,07	$\sum x^2$= 16,26

Logo temos:

$$S = \sqrt{16{,}26 - \frac{(8{,}07)^2}{5}} => \sqrt{\frac{16{,}26 - 13{,}024}{4}} => \sqrt{\frac{3{,}236}{4}} => 0{,}8\ segundos$$

Para a equipe (C), o desvio padrão na apresentada sobre a média aritmética foi de 1.628 ± 0,08.

Coeficiente de Rendimento (CR) da turma:

Para medir o coeficiente de rendimento que indica o índice de desempenho da turma de linguagem de programação C, utilizou-se dos seguintes cálculos descritos no Quadro 6.8. Primeiro listou-se as equipes, em seguida o professor atribuiu as notas para as equipes, multiplicou-se pelos tempos de cada equipe respectivamente e obteve-se o coeficiente de rendimento. Logo somamos todos os CR e todos os tempos. Pegamos o CR e dividimos pelo tempo e obtivemos o CR geral das turmas.

Quadro 6.8 - Dados de aproveitamentos

Equipes	Nota	Multiplicação	Tempo	CR
Equipe A	10,00	*	1.614s	16.14
Equipe B	9,50	*	1.618s	15.371
Equipe C	9,00	*	1.628s	14.625
Total			**4.86s**	**46.136**

Coeficiente de Rendimento (CR) = 46.136/4.86 = 94.630%

Em suma tivemos um rendimento total de 94.630% das turmas. Esse rendimento demonstra que os resultados obtidos pelos alunos no decorrer dos experimentos sobre as atividades foram positivos, ou seja, os alunos construíram o conhecimento ministrado pelo professor.

6.4 Proposta Coerente Com a Teoria

Com o objetivo de descartar uma possível coincidência de resultados, o Quadro 6.9 descreve o aproveitamento dos alunos nas disciplinas de linguagens de programação publicada na Revista Interdisciplinar de Estudos da Cognição [9]. Esses resultados demonstram que a nossa proposta está coerente com a teoria. Assim, podemos afirmar que a média de aproveitamento da turma de 2007 que utiliza experimento prático em relação a esta proposta é aceitável:

Quadro 6.9- Comparações de Aproveitamentos

Turmas	Aproveitamento %
Turma de 2002	77,80
Turma de 2003	72,77
Turma de 2004	60,90
Turma de 2005	72,70
Turma de 2006	74,40
Turma de 2007 (sem experimentos)	71,10
Turma de 2007 (com experimento)	90,30
Turma atual 2015 (com experimentos)	**94.63**

Em suma, a equipe que apresentar a melhor média de tempo será a equipe vencedora da competição. Assim, a equipe “A” obteve a média de tempo gasto para completar as tarefas 1.614s. Já equipe “B” obteve a média de tempo gasto para completar as tarefas de 1.618s e finalmente equipe “C” obteve a média de tempo gasto para completar as tarefas 1.628s. Portanto, concluímos que a equipe “A” obteve o melhor tempo e o Coeficiente de Rendimento da turma foi de 94.63%.

6.5 Avaliação Geral e Pedagógica do Sistema FDRobô

O sistema FDRobô foi avaliado por meio dos preenchimentos dos questionários, norteando a satisfação com o usuário e a importância do sistema na execução das tarefas pedagógicas. Os questionários adotados nesta pesquisa foram organizados em duas categorias:

Para a categoria usabilidade geral utilizou-se os dez princípios de usabilidade Heurística, propostos por Nielsen [35]. Esta Heurística (Anexo III) visa verificar a satisfação do aluno em relação ao sistema proposto e foi aplicada após os estudantes terem solucionado a lista de tarefas. Esta Heurística utiliza alternativa como: satisfaz, satisfaz parcialmente, e não satisfaz.

Para a categoria usabilidade pedagógica e com o propósito de dar suporte à mensuração da qualidade metodologia do ensino, utilizou o questionário de Nokelainen [37] (Anexo II). Por meio de uma escala, é medida a satisfação do usuário com a interface da aplicação e a sua eficiência pedagógica. As respostas são dispostas em uma escala *Lickert* [35] com pontuação de 1 a 5, indo de Concordo Totalmente (5 pontos) até Discordo Totalmente (1 ponto), conforme a Figura 6.9:

Figura 6.9 - Resposta do questionário em escala *Lickert*

6.5.1 Avaliação da Usabilidade Geral

Os entrevistados das equipes A, B e C, em um total de 15 alunos avaliaram a ferramenta por meio da usabilidade geral. A Figura 6.10 descreve os resultados. Foram avaliados 10 (dez) itens conforme do Quadro 6.10 e a Média Ponderada (MP) de satisfação dos alunos foi de 70% dos estudantes estão satisfeitos com a ferramenta. Já para MP = 26,25% dos estudantes apresentam uma satisfação parcial do sistema e para os alunos não satisfeitos temos MP = 3,75%. Vale ressaltar que os alunos conseguiram realizar as tarefas básicas no seu primeiro contato com um protótipo proposto.

	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Satisfaz	62,5%	100,0%	87,5%	62,5%	37,5%	87,5%	75,0%	62,5%	62,5%	62,5%
Satisfaz Pacialmente	37,5%	0,0%	12,5%	37,5%	62,5%	12,5%	25,0%	37,5%	12,5%	25,0%
Não Satisfaz	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%	25,0%	12,5%

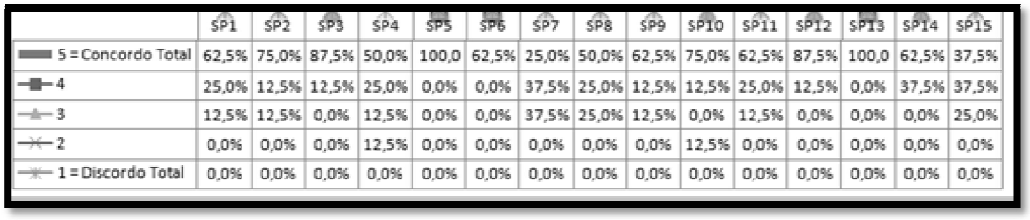
Figura 6.10 - Dados da Heurística de Usabilidade Geral

Quadro 6.10 - Legenda da Heurística de usabilidade geral

Heurística	Pergunta:
H1	Visibilidade do contexto atual do sistema
H2	Compatibilidade entre o sistema e o mundo real
H3	Controle e liberdade do usuário
H4	Padrões na interface gráfica
H5	Prevenção de erros
H6	Reconhecimento ao invés de memorização
H7	Flexibilidade e eficiência de uso
H8	Projeto estético minimalista
H9	Diagnosticar e corrigir erros
H10	Informações de ajuda e documentação

6.5.2 Avaliação da Usabilidade Pedagógica

A Usabilidade Pedagógica visa verificar se a ferramenta pode auxiliar os alunos no processo de aprendizagem, ou seja, o *framework* proposto atingiu com satisfação a expectativa do aluno em sala de aula. A Figura 6.11 descreve os resultados das equipes A, B e C, total de 15 alunos avaliaram o sistema FDRobô utilizando a escala *Lickert* com pontuação de 1 a 5 indo de Concorda Totalmente (5 pontos) até Discordo Totalmente. A Média Ponderada (MP) do item Concorda Totalmente foi de 76,66, logo se concluiu que a média ponderada de satisfação pedagógica para os usuários do sistema foi igual a 76,66%. Esse resultado demonstra que o *framework* proposto atingiu com satisfação a expectativa do aluno em sala de aula. O Quadro 6.11 descreve a legenda da Figura 6.11. Vale ressaltar que os estudantes parabenizaram o sistema proposto pela sua eficiência. Um aparticipante comentou " temos facilidade em aprender o que foi ensinado, pois todo assunto ministrado em sala de aula podem ser executado no aplicativo e sanar as dúvidas com ajuda do professor. Além disso temos a opção de revisar os conteúdo por meio do acessar a internet sem sair do programa. O aplicativo facilita muito para aqueles que nunca se depararam com a programação".



	SP1	SP2	SP3	SP4	SP5	SP6	SP7	SP8	SP9	SP10	SP11	SP12	SP13	SP14	SP15
5 = Concordo Total	62,5%	75,0%	87,5%	50,0%	100,0	62,5%	25,0%	50,0%	62,5%	75,0%	62,5%	87,5%	100,0	62,5%	37,5%
4	25,0%	12,5%	12,5%	25,0%	0,0%	0,0%	37,5%	25,0%	12,5%	12,5%	25,0%	12,5%	0,0%	37,5%	37,5%
3	12,5%	12,5%	0,0%	12,5%	0,0%	0,0%	37,5%	25,0%	12,5%	0,0%	12,5%	0,0%	0,0%	0,0%	25,0%
2	0,0%	0,0%	0,0%	12,5%	0,0%	0,0%	0,0%	0,0%	0,0%	12,5%	0,0%	0,0%	0,0%	0,0%	0,0%
1 = Discordo Total	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%

Figura 6.11 - Dados de Satisfação Pedagógica

Quadro 6.11 - Legenda da Satisfação Pedagógica

Satisfação Pedagógica (SP)	Pergunta:
SP1	Esse material pode ser entendido e usado por qualquer aluno, com pouca ou muita experiência no uso de computadores?
SP2	Foi fácil aprender a usá-lo? Você não precisou ficar pedindo muito ajuda ao professor até entender como o sistema funciona?
SP3	Você acha que o conteúdo deste material mantém a sua atenção?
SP4	Para aprender, você não usa somente o conteúdo deste material, mas usa também links para várias outras fontes, as quais têm que usar para aprender?
SP5	Quando estuda com este material, você sente que sabe mais sobre alguns tópicos do que outros? (Você aprende os outros tópicos, mas acha que se tornou um perito em alguns)
SP6	É agradável usar este material com outro colega no mesmo computador?
SP7	Após ter utilizado este material, você é avisado sobre o que está esperando saber (ou aprender)? (Ex: o programa diz: ―Após esta lição você saberá fazer perguntas em inglês).
SP8	Ao iniciar o uso deste material, mostra-se claramente porque é útil aprendê-lo? (O programa diz: ―Esta lição irá ajudá-lo a fazer frases interrogativas).
SP9	Este material lhe mostra como aplicar o conhecimento aprendido em situações de sua vida diária? (Por exemplo, calcular quantos metros quadrados tem a sala de sua casa).
SP10	Este material usa a idéia de que é melhor aprender fazendo por você mesmo? (É disponibilizado uma boa quantidade de exercícios)
SP11	Ao usar este material, sente que ele foi projetado para você? (As tarefas não são muito fáceis, nem muito difíceis).
SP12	É mais útil aprender assuntos com este material do que em sala de aula normal? (É melhor do que usar livro de estudos normal)
SP13	Você teria interesse de aprender os tópicos deste material mais profundamente? Se você tivesse mais tempo, iria querer usar mais este material para aprender um pouco mais?
SP14	Você acha que é rápido aprender um novo tópico ou recapitular um tópico anterior neste material?
SP15	Quando você usa este material, sente que tem que lembrar muitas coisas ao mesmo tempo? (Às vezes é melhor usar papel para escrever algumas anotações)

6.5.3 Análise das Avaliações

Após concluir toda a avaliação muitas recomendações foram apresentadas. Os alunos solicitaram que a tela do sumário tenha um ícone para isenção de arquivos. As equipes A e B sugeriram que fosse adicionado um módulo de vídeos visando uma melhor perspectiva do conteúdo. O módulo de atividade foi considerado interessante. A equipe C recomendou que introduzisse conceitos básicos de matemática.

Os alunos afirmaram que a colaboração entre os integrantes proporcionou a melhoria na aprendizagem, ou seja, no primeiro contato com a ferramenta alguns alunos assimilavam o conteúdo mais rápido e outros não. As equipes também relataram que a competição precisa da cooperação de todos para terem resultados positivos e que a metodologia utilizada no processo de ensino-aprendizagem de linguagem de programação C é interessante, motivadora e lúdica.

6.6 Conclusão

Neste capítulo foi descrito como o sistema FDRobô foi avaliado. Descreveu-se um conjunto de atividades que resultou no protótipo denominado de FDRobô e FDServe. A forma adotada para a construção do protótipo foi sob a forma de subsistemas. A interface Homem-máquina foi baseada na Usabilidade pedagógica e para executar as instruções no protótipo, desenvolve-se um conjunto de métodos para manipular o Robotino.

Utilizaram-se várias ferramentas, equipamentos e *softwares* baseado no estudo da arte. Na hora de integrar os componentes (Apicativo, Robotino e métodos de aprendizagem) foram diagnosticadas dificuldades de comunicação com a rede wireless do Robotino devido à baixa carga da bateria. Como se trata de uma ferramenta para atender requisitos pedagógicos foi realizado avaliações em ambientes real de aprendizagem, visando identificar melhorias em seus módulos desenvolvidos.

O protótipo foi avaliado por meio da metodologia que utiliza como base a aprendizagem cooperativa e competitiva. Primeiro foi feita a seleção do pessoal por meio de questionário autorizado pela Comissão de Ética da Universidade Federal do Amazonas (UFAM). Em seguida implementou-se um conjunto de tarefas visando avaliar o Coeficiente de Rendimento da turma e obteve-se 94.63% de aproveitamento, ou seja, esse resultado demonstra que essa ambiente esta coerente com outros trabalhos relacionados.

Na avaliação de Usabilidade geral e pedagógica a satisfação ultrapassou os 70%, tanto para avaliação de usabilidade geral quanto para avaliação pedagógica. Essas avaliações utilizaram os princípios estabelecidos pela Norma ISO 9241- Parte 10 – Princípios de Diálogo, para medir a usabilidade geral e pedagógica do protótipo.

Assim, limitações de funcionalidades que contribuíram para informar os pontos positivos e os que precisam ser melhores empregados pela ferramenta, de modo que novas versões possam ser aprimoradas e testadas de forma mais criteriosa.

Por meio destas avaliações no próximo capítulo serão descritas não só as dificuldades para o desenvolvimento do sistema proposto, mas também contribuições e trabalhos futuros que possam dar continuidade a esta pesquisa.

Capítulo 7- Considerações Finais

Este trabalho apresentou um *framework* denominado de FDRobô do tipo cliente/servidor para auxiliar a prática e a aprendizagem do aluno na disciplina linguagem de programação C. Ele é um aplicativo desenvolvido em Java e que pode ser adaptado para o ensino de outras linguagens de programação. Este aplicativo possibilita a criação de instruções visando implementar a teoria da programação C via robô móvel. Tudo pode ser feito a partir de comandos que devem ser agrupados de modo lógico.

Por meio desta arquitetura pode-se acrescentar e adaptar recursos de interatividade visando o apoio metodológico ao professor. O trabalho contribui com avaliações que demonstra que um programa de computador pode auxiliar na aprendizagem dos conteúdos ministrados em sala de aula por meio da prática e também contribui com uma arquitetura de *software* para o processo de ensino e aprendizagem de linguagens de programação.

Durante as pesquisas deste trabalho, observou-se o emprego da robótica no ambiente educacional. Foram identificadas várias oportunidades de pesquisas que melhoram a aprendizagem e interação entre o quem transmite e o quem recebe a informação por meio destes conceitos, pois nos estudos de casos desenvolvidos e testados houve uma boa aceitação da manipulação do robô por meio do *framework* proposto, porém com melhorias a serem acrescentadas e desenvolvidas. Por exemplo, adicionar comando de voz.

A arquitetura contribui ainda como um modelo de aplicações que podem ser exploradas pedagogicamente para organizar conteúdos não só para o estudo de linguagens de programação, mas servindo como base para outros assuntos da matemática ou até mesmo em disciplinas diferentes. Isso se comprova pela forma que seus módulos foram desenvolvidos com funcionalidades diferentes em um mesmo domínio específico do problema, mas que podem ser reutilizadas, devido à aplicabilidade do paradigma orientado a objetos empregado nesta arquitetura.

Deparou-se com várias dificuldades como na comunicação com o Robotino devido à falta de memória, tentou-se criar vários serviços que possibilitassem a implementação do Robotino, mas devido a sua baixa memória não foi possível. A falta de energia no

Laboratório de TV Digital inviabilizava os experimentos, pois o Robotino precisava está sempre com a carga de energia alta, caso contrário, suas funcionalidades eram afetadas. Outras dificuldades foram de mapear o comportamento do Robotino, pois as linhas de código fonte são bastante complexas. Dificuldades também na elaboração de uma interface que facilitasse a vida do professor na condução da interface gráfica. Por fim, seguiram-se detalhadamente os conceitos definidos e analisados ao longo do contexto dessa dissertação. O resultado foi um ambiente testado na plataforma do Robotino e avaliado o seu comportamento e limitações, no entanto, podendo ser superadas em trabalhos futuros. Assim o sistema proposto atingiu os objetivos seja geral e específico de forma genuína.

7.1 Dificuldades Encontradas

A maior dificuldade encontrada durante a elaboração desta dissertação de mestrado foi organizar as informações e a escrita. “Como é um trabalho acadêmico e tem que seguir as normas da universidade, a disposição das informações influencia muito, pois às vezes o trabalho pode ficar confuso e os avaliadores podem achar que está fora de ordem”. Outra dificuldade encontrada é a desmotivação como, financeiro, problemas de saúde, problemas psicológicos e outros.

7.2 Trabalhos Futuros

Os trabalhos futuros apresentados nesta seção poderão dar continuidade do aperfeiçoamento desta pesquisa, por meio de melhorias dos módulos do *framework* e oportunidades para acrescentar outros módulos que explorem os conceitos de sistemas embarcado em robôs, Objetos Digitais de aprendizagem e *e-learning*. Outra contribuição é a realização de novas pesquisas para a plataforma de celulares inteligentes. Essas possibilidades de trabalhos apresentam-se:

- Estender o módulo de atividades para o módulo “índice de linguagens”. Neste caso, cada item do índice poderá servir de acesso a uma linguagem de programação diferente;
- Migrar a aplicação para plataforma Web e *Android* tratando toda parte de interface gráfica por meio da ergonomia e usabilidade.
- Adaptar a arquitetura para atender a aplicações de educação a distância;

- Construir um repositório de atividades para cadastrar e manipular soluções.
- Adaptar a aplicação para o professor personalizar e construir sua própria aula e disponibilizar aos seus alunos para prática do aprendizado; e
- Realizar um aprofundamento teórico sobre uma teoria pedagógica a ser utilizada como referencial na construção de objetos de aprendizagem.

7.3 Comentário Final

Dentre as diversas ferramentas que auxiliam os educandos no processo de aprendizagem tem-se o computador como um grande aliado. O computador, representando as diversas ferramentas da informática e os *softwares* educativos usados na educação, torna-se cada vez mais um amplificador de potencialidades na capacitação e aperfeiçoamento de alunos, professores e das próprias instituições de ensino.

O sistema FDRobô pode ser considerado um programa educacional a partir do momento em que seja projetado por meio de uma metodologia que o contextualize no processo ensino-aprendizagem. Desse modo, mesmo um *software* detalhadamente pensado para mediar a aprendizagem pode deixar a desejar se a metodologia do professor não for adequada ou adaptada a situações específicas de aprendizagem.

Quanto ao enfoque dado à aprendizagem, o sistema FDRobô pode direcionar para uma aprendizagem lógica. Em um *software* de aprendizagem lógica a ênfase está no processo de aprendizagem, na direção que vai do sujeito que domina o saber para aquele que quer aprender. O aplicativo FDRobô tem o papel de permitir a programar de uma sequência de instruções planejadas para levar o educando ao conhecimento por meio das atividades experimentais em que o programa possa produz um ambiente com situações variadas para que o aluno as explore e construa conhecimentos por si mesmo.

Quando se desenvolveu o sistema FDRobô para apoio ao processo de aprendizagem, uma das etapas primordiais de sua produção foi definir a concepção pedagógica daqueles que estão envolvidos no seu desenvolvimento. E para isso, a socialização dos professores no laboratório de TV Digital foi essencial.

Capítulo 8- Referências Bibliográficas

[1] Abreu, A. C. B. Avaliação De Usabilidade Em Softwares Educativos. Dissertação apresentada Aplicada do Centro de Ciências e Tecnologia da Universidade Estadual do Ceará e Diretoria de Graduação e Pós-Graduação do Instituto Federal de Educação, Ciência e Tecnologia do Ceará, Fortaleza - Ceará 2010.

[2] AdaCore (2015)."The Ada Programming Language". Disponível em: <http://www.adacore.com/adaanswers/about/ada> Acesso em: 27 julho 2015.

[3] Albuquerque, W. C., 2012. O-Learning Tvdi: Um *Framework* De Tv Digital Interativa Para Manipular Objetos Digitais De Aprendizagem Relacionados Ao Ensino Da Trigonometria. Dissertação de Mestrado. Universidade Federal do Amazonas – Programa De Pós-Graduação Em Engenharia Elétrica. Manaus, Amazonas. Brasil.

[4] Arends, R. (2008). Aprender a Ensinar. Lisboa: McGrawHill, 576 p.- pág.365.

[5] Gilleanes T. A.UML 2 : uma abordagem prática,Guedes. 2. ed. - São Paulo: Novatec Editora, 488 p, 2011, Pág. 19.

[6] CEP (2015). Comitê de Ética da UFAM Disponivel em: < http://www.cep.ufam.edu.br/index.php/home >. Acesso em: 04 agosto 2015.

[7] Cybis, W.; Betiol, A. H.; Faust, R. Ergonomia e Usabilidade -Conhecimentos, Métodos e Aplicações. 2ª Edição, 422 p. Ano: 2010, p. 220-237.

[8] Eclipse. (2015). Disponível em:< https://eclipse.org/ > , Acesso em 25 Agosto 2015. Oracle(2015). Disponível em: <http://www.oracle.com/technetwork/pt/java/javase/downloads/index.html>. Acesso em: 25 agosto 2015.

[9] Fontes, C. R.; Silva, Fabio W. O. O Ensino Da Disciplina Linguagem De Programação Em Escolas Técnicas. Ciências & Cognição, v. 13, n. 2, 2008. Pág. 84-98.

[10] Festo Didactic, Disponível em:<http://www.festo-didactic.com/int-en/learning-systems/education-and-research-robots-robotino/ > Acesso em: 09 abril 2015.

[11] Fleischfresser, L. Aprendizagem baseada em problemas: Uma estratégia de aprendizagem ativa com potencial interdisciplinar na educação em Engenharia. Anais: XLII – Congresso Brasileiro de Ensino de Engenharia. Juiz de Fora: MG, 2014. Pág. 1-10

[12] Frank, B.,Krithivasan, M. H., Ciarán Mc G. (2014).“ Using Cooperative Learning to Enhance Critical Reflection”. In 2014 Frontiers in Education Conference Proceedings, October 2014.Pág. 1899-1906.

[13] Gowin, D.B.; Alvarez, M. (2005).The art of educating with V diagrams. New York: Cambridge University Press. 103 p. - Pág. 60-99.

[14] Hengholm Junior, Helio. Engenharia de Software na prática. São Paulo, Editora Novatec, 2010, 440 p. - Pág 60-120.

[15] Hermann Kopetz. Real-Time Systems: Design Principles for Distributed Embedded. Second Edition-USA/2011, 396 p. - Pág. 38.

[16] Herbert Schildt. C Completo e Total. Editora Makron, 3ª Edição, 810 p. - pp 03-70. 1997.

[17] Hsieh, J.; Chen, C.; Lin, H. Social Interaction Mining based on Wireless Sensor Networks for Promoting Cooperative Learning Performance in Classroom Learning Environment. The 6th IEEE International Conference on Wireless, Mobile, and Ubiquitous Technologies in Education. pp 219-221. 2010.

[18] ISO 1347. Human-centred design processes for interactive systems. Gènève: International Standards Organisation, pp 29.

[19] ISO9241-10:1996.Disponível em:< http://www.iso.org/iso/catalogue_detail.htm?csnumber=16882 >, Acesso em: 25 agosto 2015. pp 22.

[20] ISO 9241-171. Ergonomics of human-system interaction -- Part 171: Guidance on software accessibility. 2008. pp 22-88.

[21] Jin, Haiwei. The design of combined platform for web-based cooperative learning.Eighth IEEE International Conference on Embedded Computing; IEEE International Conference on Scalable Computing and Communic. 2009.

[22] Johnson, DW; Johnson, R. (1974). Estrutura instrucional objetivo: Cooperativa, a concorrência, ou individualista. Revisão de Pesquisas Educacionais , 44 , Pág. 213-240.

[23] Kris Jamsa, Lars Klander. Programado em C++ A Bíblia. Makron Book, 1999. 810 p. - Pág. 10.

[24] Krithivasan S., Shandilya S., Arya K., Lala K., Manavar P., Patil S. and JAIN S. (2014).“Learning by Competing and Competing by Learning: Experience from the e-Yantra Robotics Competition”. In 2014 Frontiers in Education Conference Proceedings,October 2014. pp 1-8.

[25] Lima, A. C. O. Abordagem Metodológica Híbrida Para Avaliação Da Usabilidade de Recursos de Acessibilidade Para Deficientes Visuais. Tese de Doutorado Apresentada a Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática, Campina Grande – março de 2012. Pág. 25-38.

[26] LabiUtil 2015. Disponivel em: <http://www.labiutil.inf.ufsc.br/ergolist/check.htm> acesso em: 04 agosto 20105. Pág 1-18.

[27] Lego® Mindstorms® NXT. disponivel em: < http://www.nxtprograms.com/index.html > acesso em: 05 agosto 2015.

[28] Leland B.; Alexander C. Cooperative Learning Instructional Methods for CS1: Design,Implementation, and Evaluation.ACM Trans. Comput. Educ. 13, 3, Article 10 (August 2013), 21 pages. DOI: http://dx.doi.org/10.1145/2492686.

[29] Regueras, L. M.; Verdú, E.; Muñoz, M. F.; Pérez, M. A.; Castro, J. P.; Member, IEEE, and María J. Vú. Effects of Competitive E-Learning Tools on Higher Education Students: A Case Study. Ieee Transactions on Education, vol. 52, no. 2, may 2009. pp 279-285.

[30] Lima, Adilson da Silva. UML 2.0: Do Requisito à Solução. 3. ed. São Paulo: Érica, 2008. Pág. 20-26. I.S.B.N. 9788536503776.

[31] Lin, E.; Lin, H.; Chou, C. Development of Cloud Cooperative Learning Style Scales:Applying cloud computing concept on cloud cooperative learning of information science education. IEEE International Conference on Computer Science and Service System. 2012. pp 68-71.

[32] Maia, L.; Oliveira D. GameTVD: uma proposta de arquitetura para framework de jogos 2D para TV digital. Dissertação (Mestrado em Informática), Universidade Federal do Amazonas, Manaus, 2010.

[33] Maloney, J., Resnick, M., Rusk, N., Silverman, B. e Eastmond, E. (2010) "The Scratch Programming Language and Environment", ACM Transactions on Computing Education, Volume 10, Issue 4. pp 1-15.

[34] Medeiros, T.; Brasil P.; Aranha E. (2014). Um *framework* para criação de jogos voltados para o ensino de lógica de programação. Anais: III Congresso Brasileiro de Informática na Educação (CBIE)/XXV Simpósio Brasileiro de Informática na Educação (SBIE), 2014. Pág. 1113-1117.

[35] Nielsen, Jakob. Usability Engineering. Chapter 3 (Generations of User Interfaces) (1993) Edition: Kindle. 223 p - pp.49-69.

[36] Nikolaidis S., Keren GU, Ramakrishnan R., Shah J.(2015) "Efficient Model Learning from Joint-Action Demonstrations for Human-Robot Collaborative Tasks",ACM Transactions on Computing Education.Portland, Oregon, USA. pp 189-196.

[37] Nokelainen, P. An Emprical Assessment of Pedagogical Usability Criteria for Digital Learning Material with Elementary School Students. Published by Journal of Educational Technology & Society, v.9 (2), 2006, p. 178 – 197.

[38] Oliveira, J.; João A. Gonçalves de. Apoio à Avaliação de Usabilidade na Web – desenvolvimento do USEWEB. Dissertação de Mestrado Profissional em

Computação. Instituto de Computação - Universidade Estadual de Campinas. Pág. 116. 2006.

[39] Papert, S. Logo: computadores e educação. São Paulo: Editora, Brasiliense, 220 p. pág 128-156. 2007.

[40] Piaget, J. (1970). Epistemologia Genética. Rio de Janeiro: Vozes. 280p, pág 120-137.

[41] Pressman, R. S.; Software Engineering: A Practitioner's Approach, 7 ed., McGraw Hill, 2010. 976 p. - Pág. 182-196.

[42] PUCRS. XI Desafio de Robôs. Disponivel em < http://www.pucrs.br/eventos/desafio/mariaines.php#educ > Acesso em: 28 agosto 2015.

[43] Reelsen Alexander. Play *Framework* Cookbook. 2011. Published by Packt Publishing Ltd. Livery Place Birmingham B3 2PB, UK ISBN 978-1-849515-52-8. 292 p.

[44] Ribeiro, C. M. C. Aprendizagem Cooperativa na sala de aula: Uma estratégia para aquisição de algumas competências cognitivas e atitudinais definidas pelo ministério da educação. 2006. 222 f. Dissertação (Mestrado em Biologia e Geologia para o ensino) Universidade de Trás-os-Montes e Alto Douro, Vila Real, 2006. Pág 29-57.

[45] RoboEduc. Robótica Educacional. Disponilvel em < http://www.roboeduc.com/ >, Acesso em: 22 agosto 2015.

[46] Santos, B. M. R.; Dias, N. C.; Castilho, O. V. R.; Alves, S. D. Software Educativo: Uma Ferramenta de Aprendizagem da Matemática na Educação Infantil. Revista Científica Eletrônica de Pedagogia – ISSN: 1678-300x. Ano x – número 20 – julho de 2012.

[47] Scratch. (2015). Disponível em: http://scratch.mit.edu. Acesso em: 20/04/2015.

[48] Savi R.; Wangenheim, C. G. von; Borgatto, A. Ferreti.” A Model for the Evaluation of Educational Games for teaching Software Engineering”, 25th Brazilian Symposium on Software Engineering - pp. 194-203,© 2011 IEEE. pp 194-203.

[49] Sebesta, R. W. Conceitos de Linguagens de Programação, 9ª ed., Bookman, 2011. 280 p. - Pág. 47-131.

[50] Secchi, H. A. Uma Introdução aos Robôs Móveis. Ed. Abril de 2012. 79 p. - Pág. 3-25.

[51] Silva, Marcio G.de L.; Sabaiani, H.; Menolli, A.L.A.; Busman, C.J. Avaliação de Softwares Educacionais para o Ensino Fundamental no Auxílio do Processo de Ensino–Aprendizagem. Disponível em: http://www.gied.ffalm.br/artigos/AvSwEducacional.pdf, Acesso em: 16 de junho de 2015.

[52] Smalltalk (2015). "Community and industry meet inventing the future". Disponível em: < http://www.smalltalk.org/smalltalk/TheEarlyHistoryOfSmalltalk_Abstract.html > Acesso 28 julho 2015.

[53] Sommerville, Ian. Engenharia de Software. 9º Ed. São Paulo, 529 pág: Pearson Prentice Hall, 2011. Pág. 296-307.

[54] Sumita, M. C. R.; Rajendra, R. T. H.; Jennifer S. (2013). “A Curricular Framework for Critical Infrastructure Protection Education for Engineering, Technology and Computing Majors”. In 2013 Frontiers in Education Conference Proceedings, October 2013. Oklahoma City, Oklahoma, USA. pp 1779-1781.

[55] Hart, S.; Dinh, P.; Yamokoski, J. D.; Wightman, Br. and Nicolaus R. (2014) "Robot Task Commander: A Framework and IDE for Robot Application Development", 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2014)September 14-18, 2014, Chicago, IL, USA. pp 1547-1554.

[56] Tucker, A.; Noonan, R. Linguagens de Programação: Princípios e Paradigmas. McGraw-Hill, 2009. Pág. 20-30.

[57] Usability Net. Questionnaire resources. Disponível em: <http://www.usabilitynet.org/tools/r_questionnaire.htm>. Acesso em: 03 agosto 2015.

[58] VEX. VEX Robotics. Disponilvel em: < http://www.vexrobotics.com/vexiq/ > Acesso em 30 agosto 2015.

[59] Vygotsky, L.S. (1987). Pensamento e linguagem. S. Paulo: Martins Fontes. Cáp V, 1-10.

[60] Wan-Ling Chang, Selma Šabanović.(2015) "Interaction Expands Function: Social Shaping of the Therapeutic Robot PARO in a Nursing Home", ACM Transactions on Computing Education, Portland, OR, USA.

[61] Webster, M. (2008) New World Dictionary: Online. Disponível em: < http://www.merriam-webster.com/ > acesso em: 10 julho 2015.

[62] William Stewart. Disponilvel < http://www.livinginternet.com/i/iw_unix_c.htm >. Acesso em: 25 agosto 2015.

[63] Yussiff, A.; Ahmad, W. F. W.; Oxley A. (2014). “Conceptual framework for effective E-collaboration and didactic enhancement”. in IEEE/Computer and Information Sciences (ICCOINS), 2014 International Conference on. ISBN:978-1-4799-4391-3,3-5 June 2014.

[64] Oracle. Java Software. Disponilvel em < https://www.oracle.com/java/index.html >. Acesso em: 10 agosto 2015.

Apêndice A- Publicação

PICANÇO, W. SOUZA e LUCENA JR, V. (2015). *Framework* Didático Para O Ensino De Programação Embarcada: Uma Abordagem Baseada em Técnicas De Aprendizagem Cooperativa e Competitiva" Publicado nos anais do XLIII Congresso Brasileiro de Educação em Engenharia (COBENGE-2015).

Apêndice B- Outra Publicação

ALBUQUERQUE, W. CARVALHO; SIMÕES, WALTER C. S; PICANÇO, S. WOLLACE; LIRA, ANTONIO DA F.; NETO, JOSÉ P. DE QUEIROZ. (2015) "*Framework* de TV Digital Interativa para o Ensino de Trigonometria através de Manipulação de Objetos Digitais de Aprendizagem" Artigo Aceito para publicação do I Congresso Amazônico de Computação e Sistemas Inteligentes – CACSI 2015.

Apêndice C- Termos do FDRobô e FDServe

Framework Didático	*Software* Educacional para auxilia a prática no ensino da linguagem de programação C.
FDRobô	Aplicativo cliente responsável pela interface gráfica entre o aluno e o FDserve.
FDServe	Aplicativo servido responsável em receber dados e enviar para o Robotino.
frent1	Variável que faz o Robotino se deslocar para frente.
direita1	Variável que faz o Robotino se deslocar para direita.
esquerda1	Variável que faz o Robotino se deslocar para esquerda.
costa1	Variável que faz o Robotino se deslocar para traz.
Direita45°()	Função que faz o Robotino gira em torno do seu eixo 45° a direita.
Direita90°()	Função que faz o Robotino gira em torno do seu eixo 90° a direita.
Direita135°()	Função que faz o Robotino gira em torno do seu eixo 135° a direita.
Direita180°()	Função que faz o Robotino gira em torno do seu eixo 180° a direita.
Direita225°()	Função que faz o Robotino gira em torno do seu eixo 225° a direita.
Direita270°()	Função que faz o Robotino gira em torno do seu eixo 270° a direita.
Direita315°()	Função que faz o Robotino gira em torno do seu eixo 315° a direita.
Direita380°()	Função que faz o Robotino gira em torno do seu eixo 380° a direita.
Esquerda45°()	Função que faz o Robotino gira em torno do seu eixo a esquerda 45°.

Esquerda90°()	Função que faz o Robotino gira em torno do seu eixo a esquerda 90°.
Esquerda135°()	Função que faz o Robotino gira em torno do seu eixo a esquerda 135°.
Esquerda180°()	Função que faz o Robotino gira em torno do seu eixo a esquerda 180°.
Esquerda225°()	Função que faz o Robotino gira em torno do seu eixo a esquerda 225°.
Esquerda315°()	Função que faz o Robotino gira em torno do seu eixo a esquerda 315°.
Esquerda380°()	Função que faz o Robotino gira em torno do seu eixo a esquerda 380°.
Funções Robotino	Paleta responsável em mostra as funções em graus do Robotino.
Variáveis Robotino	Paleta responsável em mostra as variáveis particular do Robotino.
FDRobô	Framework Didático e Robô.
FDServe	Framework Didático e Servidor

Anexo I- Satisfação Pedagógica

Olá!

As perguntas deste questionário são destinadas a avaliar a facilidade de uso do *software* que você acabou de utilizar. Fique à vontade em responder, pois o que está sendo avaliado é este material e não você.

1. (SP1) Esse material pode ser entendido e usado por qualquer aluno, com pouca ou muita experiência no uso de computadores?

2. (SP2) Foi fácil aprender a usá-lo? Você não precisou ficar pedindo muito ajuda ao professor até entender como o sistema funciona?

3. (SP3) Você acha que o conteúdo deste material mantém a sua atenção?

4. (SP4) Para aprender, você não usa somente o conteúdo deste material, mas usa também links para várias outras fontes, as quais têm que usar para aprender?

5. (SP5) Quando estuda com este material, você sente que sabe mais sobre alguns tópicos do que outros? (Você aprende os outros tópicos, mas acha que se tornou um perito em alguns)

6. (SP6) É agradável usar este material com outro colega no mesmo computador?

7. (SP7) Após ter utilizado este material, você é avisado sobre o que está esperando saber (ou aprender)? (Ex: o programa diz: —Após esta lição você saberá fazer perguntas em inglês).

8. (SP8) Ao iniciar o uso deste material, mostra-se claramente porque é útil aprendê- lo? (O programa diz: —Esta lição irá ajudá-lo a fazer frases interrogativas).

9. (SP9) Este material lhe mostra como aplicar o conhecimento aprendido em situações de sua vida diária? (Por exemplo, calcular quantos metros quadrados tem a sala de sua casa).

10. (SP10) Este material usa a idéia de que é melhor aprender fazendo por você mesmo? (É disponibilizado uma boa quantidade de exercícios)

11. (SP11) Ao usar este material, sente que ele foi projetado para você? (As tarefas não são muito fáceis, nem muito difíceis).

12. (SP12) É mais útil aprender assuntos com este material do que em sala de aula normal? (É melhor do que usar livro de estudos normal)

13. (SP13) Você teria interesse de aprender os tópicos deste material mais profundamente? Se você tivesse mais tempo, iria querer usar mais este material para aprender um pouco mais?

14. (SP14) Você acha que é rápido aprender um novo tópico ou recapitular um tópico anterior neste material?

15. (SP15) Quando você usa este material, sente que tem que lembrar muitas coisas ao mesmo tempo? (Às vezes é melhor usar papel para escrever algumas anotações)

Anexo II- Heurística, Avaliação Geral.

Aplicação da heuristic:	**Data / /**	
Nome:	**Início**	**Fim**
Avaliador:	**Controle de vídeo**	
Heurística 1 (H1): Visibilidade do contexto atual do sistema		
Verificação: Os usuários são mantidos informados sobre o progresso do sistema com apropriada realimentação em tempo razoável?		
Resposta: ( ) Satisfaz ( ) Satisfaz Pacialmente ( ) Não Satisfaz		
Heurística 2 (H2): Compatibilidade entre o sistema e o mundo real		
Verificação: É utilizado no sistema conceitos e linguagem familiar com o usuário em vez de termos orientados ao sistema/ O sistema utiliza convenções do mundo real, exibindo informações com uma ordem lógica e natural?		
Resposta: ( ) Satisfaz ( ) Satisfaz Pacialmente ( ) Não Satisfaz		
Heurística 3 (H3): Controle e liberdade do usuário		
Verificação: Os usuários têm opções (desfazer ou cancelar) de sair ou reverter situações em que não sejam desejáveis no sistema, como um erro na escolha de uma função do sistema, sem que haja perda de funcionalidade?		
Resposta: ( ) Satisfaz ( ) Satisfaz Pacialmente ( ) Não Satisfaz		
Heurística 4 (H4): Consistência e padrões		
Verificação: O sistema prove consistência a padrões?		
Resposta: ( ) Satisfaz ( ) Satisfaz Pacialmente ( ) Não Satisfaz		
Heurística 5 (H5): Prevenção de erros		
Verificação: O sistema prove mecanismos de prevenção de ocorrência de erro?		
Resposta: ( ) Satisfaz ( ) Satisfaz Pacialmente ( ) Não Satisfaz		
Heurística 6 (H6): Reconhecimento ao invés de memorização		
Verificação: O sistema prove mecanismos em que as ações, comandos e elementos da interface sejam acessíveis ao usuário, sem que este precise memorizar ações e procedimentos na aplicação?		
Resposta: ( ) Satisfaz ( ) Satisfaz Pacialmente ( ) Não Satisfaz		
Heurística 7 (H7): Flexibilidade e eficiência de uso		
Verificação: O sistema propicia mecanismos variados para execução de uma mesma tarefa?		
Resposta: ( ) Satisfaz ( ) Satisfaz Pacialmente ( ) Não Satisfaz		
Heurística 8 (H8): Projeto estético minimalista		
As janelas e interface de diálogo do sistema contêm informações irrelevantes ou desnecessárias?		
Resposta: ( ) Satisfaz ( ) Satisfaz Pacialmente ( ) Não Satisfaz		
Heurística 9 (H9): Diagnosticar e corrigir erros		
Verificação: As mensagens de erro são expressas em linguagem clara, indicando precisamente o problema e sugerindo soluções?		
Resposta: ( ) Satisfaz ( ) Satisfaz Pacialmente ( ) Não Satisfaz		
Heurística 10 (H10):: Informações de ajuda e documentação		
O sistema fornece ao usuário documentação de ajuda e informações de ajuda?		
Resposta: ( ) Satisfaz ( ) Satisfaz Pacialmente ( ) Não Satisfaz		

Anexo III- Ferramentas de Desenvolvimento

Nº	Descrição
01	Sistema Operacional Windows 7 *Home Premium* Plataforma PC de 64 bits
02	Computador com Processador AMD *Turion II Dual Core Mobile* M500 2.2 Ghz, 4 Gb de memória *RAM*
03	Linguagem Java
04	*Kit* de Desenvolvimento Java (JDK)
05	*Software Astar Pro*
06	Plataforma de Aplicações Eclipse
07	Plugin JavaFX
08	Sistema Operacional Linux Fedora