UNIVERSIDADE FEDERAL DE SÃO JOÃO DEL-REI
DEPARTAMENTO DE EDUCAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO
PROCESSOS SÓCIO-EDUCATIVOS E PRÁTICAS ESCOLARES

CONSTRUCIONISMO DE PAPERT E ENSINO-APRENDIZAGEM DE PROGRAMAÇÃO DE COMPUTADORES NO ENSINO SUPERIOR

Márcio Roberto de Lima

SÃO JOÃO DEL-REI
MINAS GERAIS – BRASIL
AGOSTO DE 2009

CONSTRUCIONISMO DE PAPERT E ENSINO-APRENDIZAGEM DE PROGRAMAÇÃO DE COMPUTADORES NO ENSINO SUPERIOR

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação: Processos Sócio-Educativos e Práticas Escolares, como requisito parcial para obtenção do título de Mestre em Educação.

Mestrando: Márcio Roberto de Lima
Orientador: Prof. Dr. Murilo Cruz Leal

UFSJ
MINAS GERAIS
AGOSTO DE 2009

UFSJ
UNIVERSIDADE FEDERAL
DE SÃO JOÃO DEL-REI

MÁRCIO ROBERTO DE LIMA

CONSTRUCIONISMO DE PAPERT E ENSINO-APRENDIZADEM DE PROGRAMAÇÃO DE COMPUTADORES NO ENSINO SUPERIOR

Banca Examinadora

Prof. Dr. Murilo Cruz Leal - UFSJ Orientador

Prof. Dr. Luiz Henrique Andrade Correia - UFLA

Prof. Dr. Laerthe de Moraes Abreu Junior - UFSJ

AGOSTO DE 2009

À Regina e Lara,
minha feliz família,
que se fazem luzes em meu
caminho e dão sentido a minha
vida, dedico este trabalho.

AGRADECIMENTOS

Ao Pai do Céu fonte de amor, paz e bem.

Aos fiéis amigos do plano invisível, companheiros e orientadores de mais essa jornada, que não me furtaram a presença e a inspiração nos momentos de produção.

À Universidade Federal de São João del-Rei, que pela segunda vez em minha trajetória acadêmica, ofereceu-me um ambiente satisfatório para a continuidade de minha formação.

Ao meu orientador, Professor Dr. Murilo Cruz Leal, por ter aceitado o desafio empreendido, pela autonomia a mim concedida, pelo respeito com meu estilo de produção, pela competência, companheirismo e pelos inúmeros e descontraídos momentos de apoio, dedicação e aprendizado em conjunto.

A minha mamãe e ao meu papai, Marlete Mayrinck de Lima e Roberto Geraldo de Lima, exemplos de caráter e pelo suporte na educação de minha menina Lara. Sem eles este trabalho não teria sido concluído.

A minha esposa, Regina Celi Ferreira de Lima, que se mostra companheira e incentivadora em todos os momentos. Obrigado, por ter compreendido minha necessidade de dedicação e concentração na elaboração deste trabalho.

A minha menina Lara, que não me furta o sorriso e alegria (fontes de energia), mesmo com o pouco tempo disponível para nossas travessuras (*Hey, ho! Let´s go!*).

Aos meus professores do Programa de Mestrado em Educação da UFSJ, que me mostraram caminhos e me deram condições para neles aprender a andar e, quem sabe um dia, até mesmo correr! Obrigado pela receptividade, compreensão e confiança.

Aos professores Doutores Laerthe Abreu Júnior (UFSJ) e Luiz Henrique de Andrade Correia (UFLA), de quem pude receber contribuições significativas durante o Exame de Qualificação e também por terem aceitado o convite para comporem a Banca Examinadora da defesa da dissertação.

À Professora Doutora Cláudia Mariza Braga e ao professor Adriano Zanetti, responsáveis pelas revisões de Português e de Inglês, pela disposição, capricho e competência.

A Simone Rocha Gonçalves (secretária do Programa de Mestrado em Educação da UFSJ) e Márcia Magalhães dos Santos Lima, a “Marcinha” (secretária da Pró-Reitoria de Ensino da UFSJ), colaboradoras competentes e dedicadas, sempre disponíveis a nos atender e ajudar.

Aos professores Doutores Heitor Antônio Gonçalves e Carlos Henrique de Souza Gerken, que gentilmente se dispuseram a compor a suplência da Comissão Examinadora da dissertação.

A toda família Lima, sempre presente e incentivadora.

A todos os amigos de mestrado, especialmente a Camila, Maricéa, Apolliane, Patrícia, Maria Jussara, Saly, Valcíria e Rebeca, pelo respeito e confiança no "menino engenheiro". Obrigado por terem me ajudado durante a adaptação à nova área de conhecimento e por fazerem nossos encontros mais produtivos e descontraídos.

A todos os meu inspiradores, especialmente a Joey Ramone (*in memoriam – Jeff Hyman*),figura impar do cenário musical e ícone da minha "eterna" juventude, por ter ritmado a elaboração deste trabalho. *Gabba, Gabba, Hey! Gabba, Gabba, Hey!*

Aos alunos e professores que concederam preciosos depoimentos, indispensáveis a concretização deste trabalho.

A todos aqueles que, de alguma forma, cooperaram para a realização deste trabalho: muito obrigado!

Na educação, a mais elevada marca do sucesso não é ter imitadores, mas inspirar outros a irem além.

Seymour Papert

SUMÁRIO

LISTA DE FIGURAS **8**

LISTA DE TABELAS **11**

LISTA DE SIGLAS E ABREVIATURAS **12**

RESUMO **13**

ABSTRACT **14**

INTRODUÇÃO **15**

CAPITULO I

CONSOLIDAÇÃO DAS TECNOLOGIAS DA INFORMAÇÃO E COMUNICAÇÃO: EMERGÊNCIA DE UMA PESQUISA **19**

1.1– Educação em Nível Superior: a reação às demandas emergentes 23

1.2– Estrutura de um curso de Sistemas de Informação 26

CAPÍTULO II

COMPUTADORES NA EDUCAÇÃO: POSSIBILIDADES E CONCEITOS **28**

2.1– Máquinas de ensinar 28

2.2– Instrucionismo: o computador como uma máquina de ensinar 31

2.3– Construcionismo: o computador como ferramenta de aprendizagem 33

2.4– Linguagens de programação: meio de expressão de idéias 36

2.5– A atividade de programação de computadores 42

2.6– Subsídios teóricos do construcionismo 50

CAPÍTULO III

LOGO: CONCRETIZAÇÃO DO CONSTRUCIONISMO 56

3.1– Origem e evolução 56

3.2– LOGO na perspectiva computacional 61

3.3– LOGO e suas possibilidades pedagógicas 71

CAPÍTULO IV

ALUNOS E PROFESSORES: AGENTES DO CONHECIMENTO NA ABORDAGEM CONSTRUCIONISTA 74

4.1– Implicações do uso de computadores e linguagens de programação na docência 74

4.2– Ação & Reflexão: indicativos para a atuação do professor 78

4.3– Alunos e a aprendizagem de programação de computadores 81

CAPÍTULO V

LOGO: PENSO E EXISTO 89

5.1– O perfil da turma pesquisada 91

5.2– Análise dos dados: as falas dos alunos 96

5.2.1– A visão dos alunos sobre o ensino-aprendizagem de programação 97

5.2.2– Os sentimentos de frustração e desânimo 99

5.2.3– Os alunos e os fatores motivacionais ligados à aprendizagem de programação 100

5.2.4– À espera de um professor construcionista 105

5.2.5– LOGO: uma alternativa para o ensino de programação 109

5.3– As falas dos professores 117

5.3.1– Participação na aprendizagem do aluno 118

5.3.2– Preparo para a docência 121

5.3.3– O processo de ensino-aprendizagem de programação 123

CONSIDERAÇÕES FINAIS 126

REFERÊNCIAS BIBLIOGRÁFICAS 132

ANEXO 1 138

ANEXO 2 139

LISTA DE FIGU*R*AS

Figura 1.1 Capa da edição histórica da *Popular Electronics* com o anúncio do Altair 8800..... 20

Figura 1.2 TIC, ano de surgimento e tempo gasto para que 50 milhões de pessoas as utilizassem..... 21

Figura 1.3 Receita Operacional Líquida (Bilhões de Reais) por atividades dos Serviços 23

Figura 1.4 Cursos da área de computação no final do ano de 2006 25

Figura 1.5 Distribuição de unidades curriculares por núcleo de acordo com a carga horária no curso de SI..... 27

Figura 2.1 Abordagem instrucionista de ensino..... 31

Figura 2.2 Tela do programa Eletronic Workbench simulando um circuito trifásico e a leitura de duas dessas em um osciloscópio..... 32

Figura 2.3 Linhas de ensino-aprendizagem usando o computador..... 35

Figura 2.4 Etapas do processamento de dados..... 36

Figura 2.5 Ambiente de programação Turbo Pascal versão 7 da Borland Internacional..... 38

Figura 2.6 A ferramenta BloodShed Dev-Pas regida pela GNU..... 38

Figura 2.7 Algoritmo escrito sob a forma de programa usando PASCAL..... 41

Figura 2.8 Caricatura do método transmissivo de ensino..... 42

Figura 2.9 Interação aluno-computador-professor estabelecida na atividade de programação..... 43

Figura 2.10 Esquema do processo de DERD..... 45

Figura 2.11 A espiral do conhecimento seguindo o processo de DERD de programas de computador..... 48

Figura 2.12 Construção de um programa com a metáfora do crescimento da cebola 49

Figura 2.13	*Empty graduates,* por Ibnelson	54
Figura 3.1	Papert e o robô Tartaruga	60
Figura 3.2	O ambiente do SuperLogo: a esquerda a "janela gráfica" e a direita a "janela de comandos"	61
Figura 3.3A	Passo 1 da construção do retângulo no modo direto do SuperLogo	63
Figura 3.3B	Passo 3 da construção do retângulo no modo direto do SuperLogo	63
Figura 3.3C	Passo 3 da construção do retângulo no modo direto do SuperLogo	63
Figura 3.3D	Passo 4 da construção do retângulo no modo direto do SuperLogo	63
Figura 3.3E	Passo 5 da construção do retângulo no modo direto do SuperLogo	60
Figura 3.3F	Passo 6 da construção do retângulo no modo direto do SuperLogo	60
Figura 3.3G	Passo 7 da construção do retângulo no modo direto do SuperLogo	60
Figura 3.3H	Passo 7 da construção do retângulo no modo direto do SuperLogo	64
Figura 3.4	Modo de edição de procedimentos no SuperLogo com o programa quadrado de lado 100	65
Figura 3.5	Modo direto reconhecendo o novo termo "quadrado"	66
Figura 3.6	O programa quadrado como conceito de estrutura de repetição	67
Figura 3.7	O programa quadrado usando passagem de parâmetro e repetição	68
Figura 3.8	O procedimento quadrado como subprocedimento no procedimento giraquadrado	68
Figura 3.9	Construção de uma espiral com recursão	69
Figura 4.1	Configuração da ação reflexiva do professor	77
Figura 5.1	Conhecimento adquirido em programação	92
Figura 5.2	A importância das UC de programação	92
Figura 5.3	Motivação dos alunos durante os cursos de programação	93
Figura 5.4	A importância atribuída ao professor de cursos de programação	94
Figura 5.5	A importância do envolvimento do aluno durante os cursos de programação	95
Figura 5.6	Comparação entre dedicação e aproveitamento dos alunos em UC de programação	95

Figura 5.7 Um bairro em LOGO..... 114

Figura 6.1 O Linux Educacional 3.0 do MEC com a ferramenta KTurtle instalada: Linguagem LOGO..... 131

LISTA DE TABELAS

Tabela 1.1	Criação de cursos superiores relacionados à área de computação.....	24
Tabela 3.1	Comandos básicos do LOGO seguidos de suas descrições e exemplos...	62
Tabela 3.2	Representação do processamento do programa Espiral.....	70
Tabela 5.1	Ocorrências de respostas referentes ao grau de motivação.....	93

LISTA DE SIGLAS E ABREVIATURAS

ACE	*Automatic Computer Engine*
CAI	*Computer–Aided Instruction*
DERD	Descrição, Execução, Reflexão e Depuração
EAD	Educação a distância
ENIAC	*Eletronical Numerical Integrator and Compututer*
GNU	*General Public Licence*
IBGE	Instituto Brasileiro de Geografia e Estatística
IBM	*International Business Machines*
LPC	Linguagem de Programação de Computadores
MIT	*Massachusetts Institute of Technology*
MITS	*Micro Instrumentation and Telemetry Systems*
PC	*Personal Computer*
PUC	Pontifícia Universidade Católica
TIC	Tecnologias de Informação e Comunicação
UC/UCs	Unidade Curricular/Unidades Curriculares
ZDP	Zona de Desenvolvimento Proximal
WWW	*World Wide Web*

RESUMO

A sociedade moderna convive cotidianamente com a evolução das Tecnologias da Informação e Comunicação (TIC). Esse panorama provocou a adoção do computador como elemento importante e, às vezes, indispensável para a maioria das áreas de conhecimento. Assim, o presente trabalho trata da inserção do computador na educação superior e enfoca, especificamente, o processo de ensino-aprendizagem de programação de computadores sob a perspectiva construcionista de Seymour Papert. A pesquisa empreendida buscou proporcionar uma melhor compreensão da dinâmica educacional que envolve a programação de computadores, levando em conta as ferramentas de *software* e os sujeitos envolvidos: alunos e professores. O objetivo específico do estudo foi verificar a pertinência da linguagem de programação LOGO enquanto ferramenta facilitadora da aprendizagem de programação em um curso de Sistemas de Informação de uma Universidade do interior do Estado de Minas Gerais. A abordagem utilizada para o desenvolvimento deste trabalho foi de natureza qualitativa, tendo com instrumento de coleta de dados o questionário e a entrevista semi-estruturada. Os dados coletados foram transcritos e submetidos à análise de conteúdo. Os resultados alcançados apontam o LOGO como uma ferramenta de fácil apropriação e adequada para ser trabalhada em cursos de introdução à programação de computadores, em conexão com outras linguagens. Do ponto de vista dos alunos, percebeu-se que o principal fator motivacional de suas aprendizagens está ligado à atuação do professor das unidades curriculares de programação. Desse profissional espera-se não apenas a competência técnica, mas também a pedagógica, no sentido de propiciar a didatização dos conteúdos. Para os professores, destaca-se a necessidade constante de pesquisa sobre os temas explorados, a reflexão sobre as estratégias de ensino e a vivência na área, visando a efetivação de um processo eficaz de aprendizagem.

Palavras-chave: LOGO, programação, computadores, construcionismo, ensino-aprendizagem.

ABSTRACT

Modern society is daily connected with the developments of Information and Communication Technologies (ICT). This reality has led to the use adoption of the computer as an important and, sometimes, indispensable tool for most areas of knowledge. The present work deals with the insertion of the computer in higher education and specifically focuses on the process of teaching and learning of computer programming under the constructionist perspective of Seymour Papert. To that end, the undertaken research aimed to provide a better understanding of the dynamics surrounding the educational programming of computers, taking into account the software tools and the subjects involved: students and teachers. The specific objective of the study was to evaluate the appropriateness of the programming language LOGO as a tool facilitating the learning of programming in a course of Information Systems of University of the State of Minas Gerais. The approach used for the development of this work was qualitative, and with data collection instrument through a questionnaire and semi-structured interview. The data collected were transcribed and subjected to content analysis. The results show the LOGO as a tool for easy and proper ownership for the ongoing work of introduction to computer programming in connection with other languages. From the students' perspective, it is noticeable that the main motivational factor in their learning is closely linked with the role of teacher in both syllabus and curriculum design. Hence, not only are such professionals expected to show both technical and pedagogical competence but also their ability to facilitate the implementation of the didactic content. For teachers, there is a need for research on the topics explored, reflecting on the strategies of education and experience in the area so that an effective learning process takes place.

Keywords: LOGO, programming, computers, constructionism, education and learning.

INTRODUÇÃO

Esta pesquisa trata da inserção do computador na educação e das implicações pedagógicas decorrentes, orienta-se pela perspectiva construcionista e pelo potencial pedagógico e computacional da linguagem LOGO, desenvolvida por Seymour Papert. Isso equivale a dizer que se encara o computador enquanto ferramenta de aprendizagem, em que os alunos, auxiliados por um profissional da educação, podem conceber a solução de problemas segundo uma linguagem de programação e, a partir disso, construírem e aprimorarem seus conhecimentos. O objetivo da presente pesquisa é compreender se (e como) a linguagem de programação LOGO se constitui uma ferramenta facilitadora da aprendizagem de programação de computadores em cursos de Nível Superior, especificamente, no de Sistemas de Informação.

Assim, esse estudo tem como ambiente de investigação um curso de Bacharelado em Sistemas de Informação, onde o uso do computador é recurso indispensável. O Curso visa à formação de recursos humanos para automação de sistemas de informação e é baseado nas recomendações da Sociedade Brasileira de Computação, reunindo em seu conteúdo programático enfoques teóricos e práticos em tecnologia da computação e administração.

O trabalho concentra-se nos aspectos relacionados à tecnologia da computação, especificamente no núcleo de unidades curriculares relacionados à programação de computadores. Destaca-se o uso das Linguagens de Programação de computadores (LPC) no meio educacional como forma de expressão criativa, instrumento de reflexão e solução de problemas.

Foram eleitos como objetos de estudo os elementos envolvidos com as disciplinas de programação de computadores: alunos, professores e ferramental de *software*. Professores e alunos são encarados como agentes da dinâmica de ensino-aprendizagem em um ambiente que tem o computador como ferramenta.

A motivação do estudo é uma melhor compreensão do processo de ensino-aprendizagem de programação de computadores, buscando fornecer subsídios para sua melhoria. Conforme apresentado por Martins e Correia (2003), Rodrigues Júnior (2002), Schultz (2003), Chaves de Castro *et al* (2003), Delgado *et al* (2004), Maltempi e Valente (2000) e Petry (2005), os resultados obtidos nas unidades curriculares que envolvem programação de computares são insatisfatórios, constatando-se: baixo nível de aprendizagem, desestímulo, evasão e reprovação. A fim de buscar o entendimento das razões desse quadro, busca-se a compreensão das seguintes questões:

A. Quais os fatores que levam a motivação/desmotivação dos alunos no processo de construção do conhecimento pelo computador via linguagem de programação?
B. O uso da linguagem LOGO aliada às tradicionalmente utilizadas (PASCAL, C, JAVA etc.) nas unidades curriculares de programação pode favorecer o processo de ensino-aprendizagem?
C. Os professores das unidades curriculares de LPC são conscientes de sua participação na construção da aprendizagem de seus alunos? Estão preparados para o exercício docente? Quais são as suas concepções sobre o exercício pedagógico ligado ao ensino-aprendizagem de programação de computadores?

O trabalho empreendido é de natureza qualitativa e exploratória. Sua realização contou com um grupo de 21 alunos formandos do curso de Sistemas de Informação da “Universidade Beta”, instituição particular localizada no interior de Minas Gerais. Com o objetivo de se especificar o perfil dos alunos participantes, foi aplicado um questionário diagnóstico e, a partir dele, elaborou-se uma estratégia para a aplicação da linguagem de programação LOGO como instrumento de formalização dos conceitos de programação de computadores. Desenvolveu-se uma unidade curricular em regime presencial, a qual promoveu uma revisão de conceitos de programação sob a perspectiva da linguagem LOGO. O aporte teórico do curso contemplou os conceitos ligados à Linguagem e também os relacionados ao uso de computadores na Educação. O curso teve duração de 40 horas e foi realizado no período de 07 de Agosto a 05 de Dezembro de 2008. Após o término da unidade curricular, os alunos com maior nível de adesão e

entusiasmo, em número de seis, foram convidados a participar desta pesquisa; desses, cinco atenderam ao pedido e foram entrevistados, sendo três alunas e dois alunos. A fim de buscar ampliar a compreensão do processo de ensino-aprendizagem de programação de computadores, também foram realizadas entrevistas com professores da área, docentes na "Universidade Beta". As percepções e concepções dos alunos e professores são o principal objeto da presente pesquisa.

O trabalho está organizado em seis capítulos. No primeiro, é apresentada uma visão panorâmica do estabelecimento e uso das Tecnologias de Informação e Comunicação (TIC) em diversos campos da atividade humana, com destaque para a Educação. Neste sentido, o capítulo inclui uma visão da estrutura do curso de Sistemas de Informação, cenário deste trabalho.

O segundo capítulo desenvolve uma revisão bibliográfica que privilegia as possibilidades do uso de computadores na educação, a compreensão das atividades cognitivas associadas ao uso de linguagens de programação e a perspectiva construcionista desenvolvida por Seymour Papert.

O capitulo três é dedicado à linguagem de programação LOGO. Nele, são apresentadas as origens da Linguagem, suas possibilidades computacionais e pedagógicas. Assim, são introduzidos os comandos básicos da linguagem e o ambiente de programação em LOGO, por intermédio do *software* SuperLogo. Apresenta-se uma série de exemplos de uso deste *software* como suporte ao ensino de conceitos de programação, tais como: atribuições, estruturas de repetição e condicionais, passagem de parâmetros etc.

No quarto capítulo, trata-se do processo de ensino-aprendizagem de linguagens de programação, norteado por uma perspectiva reflexiva. São introduzidas as implicações diretas dessa dinâmica para os sujeitos envolvidos: alunos e professores. Apresenta também alguns trabalhos assemelhados a este.

O quinto capítulo apresenta a pesquisa de campo, com ênfase para a análise e discussão dos dados obtidos. Além de buscar as respostas das questões de pesquisa, nele traça-se o perfil dos sujeitos participantes.

Finalmente, no sexto capítulo, são feitas as considerações finais, onde se procura evidenciar os resultados da pesquisa empreendida e algumas possibilidades para estudos futuros.

CAPÍTULO I

CONSOLIDAÇÃO DAS TECNOLOGIAS DA INFORMAÇÃO E COMUNICAÇÃO: EMERGÊNCIA DE UMA PESQUISA

As pessoas que criam novas idéias tecnológicas
não as fazem para as crianças. Com freqüência, fazem-nas
para a guerra, mantêm as mesmas em locais secretos
e mostram-nas de modo distanciado.
Seymour Papert

O cotidiano da sociedade contemporânea reflete permanentemente o rápido desenvolvimento tecnológico em curso. Neste novo panorama, a presença das Tecnologias de Informação e Comunicação (TIC) proporciona, a cada novo ciclo de seu desenvolvimento, outras formas de lidar com questões do conhecimento, como por exemplo, a indexação de informações para consulta em um site de Internet.

Lévy (1999) aponta o surgimento dos primeiros computadores programáveis e capazes de armazenar dados na Inglaterra e nos Estados Unidos, ainda na década de 40, sendo seu uso reservado a militares. Em sintonia com o autor, Marque e Motoyana (1994) apresentam o **ENIAC** (*Eletronical Numerical Integrator and Compututer*), utilizado na Segunda Grande Guerra, como o precursor dos computadores.

Duas décadas depois, os computadores passaram a ser difundidos a outros segmentos da sociedade. Apesar dos avanços em sua arquitetura inicial, os computadores ainda eram equipamentos de custo elevado, pouco amigáveis do ponto de vista operacional, ocupavam enormes espaços físicos e ofereciam pouca confiabilidade operacional (a queima de uma válvula, por exemplo, levava à alteração de resultados processados).

> Os computadores ainda eram grandes máquinas de calcular, frágeis, isoladas em salas refrigeradas, que cientistas em uniformes brancos alimentavam com cartões perfurados e que de tempos em tempos cuspiam listagens ilegíveis. A informática servia aos cálculos científicos,

> às estatísticas dos Estados e das grandes empresas ou a tarefas pesadas de gerenciamento (folhas de pagamento etc.). (LÉVY, 1994, p.31).

Nota-se que a partir da década de 70, com o advento do microprocessador[1], as TIC passaram a protagonizar transformações sociais, econômicas, políticas, culturais e ambientais, mostrando-se como elemento chave da sociedade moderna. Em janeiro de 1975, a revista *Popular Electronics* (figura 1.1) anunciava nos Estados Unidos o lançamento do **Altair 8800** da empresa MITS (*Micro Instrumentation and Telemetry Systems*). Originalmente, o Altair 8800 foi comercializado pela revista no formato de um kit alternativo aos modelos comerciais, o que marcou na história o surgimento do computador para o uso pessoal.

Figura 1.1 – Capa da edição histórica da Popular Electronics com o anúncio do Altair 8800.
Fonte: http://en.wikipedia.org/wiki/File:Popular_Electronics_Cover_Jan_1975.jpg acesso em 16/02/2009.

Em 12 de agosto de 1981 foi lançado o IBM PC (*Industrial Business Machines Personal Computer*). Com ele começava a popularização dos computadores, que antes só eram acessíveis às grandes empresas e órgãos governamentais. Esse

[1] Microprocessador é um dispositivo eletrônico que possui função de cálculo lógico e aritmético, gerenciamento de memória e controle de entrada e saída de informações em um computador. Sua origem remonta a 1971, quando a *Intel Corporation* lançou no mercado o microprocessador 4004.

movimento foi acompanhado do desenvolvimento da informática amigável, por meio de *softwares* que proporcionavam a operacionalidade do PC de forma descomplicada, por exemplo: uso de telas gráficas (meados da década de 80) com opções comandadas por dispositivos sensório-motores (*mouse*).

A década de 90 revelou uma grande massificação das TIC. Acompanhado da queda dos preços dos microcomputadores, popularização de redes de computadores, avanços significativos nos *softwares e hardware,* um novo fenômeno começou a se estruturar em escala global: a rede mundial de computadores, ou *World Wide Web* (WWW).

A figura 1.2 apresentada abaixo, mostra algumas TIC, o ano de sua disponibilização e também o tempo necessário para que elas começassem a ser popularizadas. Comparando as tecnologias ligadas à informática (PC e Internet) às demais, nota-se sua rápida adoção. Isso pode ser atribuído a fatores inerentes às demandas contemporâneas tais como: auxílio à tomada de decisão, aumento da quantidade de informações a serem gerenciadas, eficácia e rapidez em cálculos complexos, favorecimento na comunicação através das redes de computadores, entre outros. Recentemente, esses argumentos foram reforçados pelos dados do Suplemento de Produtos e Serviços da Pesquisa Anual de Serviços – PAS 2006 – do Instituto Brasileiro de Geografia e Estatística (IBGE), que mostram o crescimento dos serviços de informação, os quais geraram receita de R$ 137,3 bilhões, em 2006, contra R$ 129,2 bilhões em 2005.

Figura 1.2 – TIC, ano de surgimento e tempo gasto para que 50 milhões de pessoas as utilizassem.
(Adaptado de *National Center for Policy Analisis* (2000))

Lévy (1999) explicita que a WWW estabeleceu o ciberespaço e, conseqüentemente, a cibercultura. O ciberespaço constitui um novo meio de comunicação baseado na interconexão de computadores a nível mundial e a cibercultura representa um conjunto de técnicas, práticas, atitudes, modos de pensamento e valores que se instituíram no ciberespaço. No ciberespaço passou a ser possível o acesso à distância aos recursos de um computador, a troca de arquivos de forma simplificada, o envio de mensagens de forma sincrônica (por exemplo: *Windows Messenger*) ou assincrônica (correio eletrônico), conferências eletrônicas em tempo real (vídeo conferência), o estabelecimento dos negócios e comércio eletrônicos, transmissão de vídeo/som sob demanda e muitas outras possibilidades que surgem a todo o momento em nosso cotidiano.

Notadamente, a adoção das tecnologias digitais possibilita modificações nas formas de trabalho das organizações, proporcionando novas maneiras de "fazer" e, primordialmente, de se pensar o "fazer" (Moraes, 1997). Um exemplo dessa realidade são os empreendimentos tradicionais que incorporaram as transações virtuais a seu ramo de atuação. Indica-se como exemplo, o surgimento das chamadas "empresas pontocom", que têm na internet a plataforma de negociação direta seus clientes (lojas virtuais, *internet banking*, sítios de leilões etc.). Laudon e Laudon (2007) fornecem dados de que em 2005 mais de 40 milhões de empresas tinham um sítio registrado, além disso, cinco milhões de norte-americanos compram algo na Internet e outros 19 milhões pesquisam na rede um produto.

O Suplemento de Produtos e Serviços da Pesquisa Anual de Serviços – PAS 2006 – elaborado pelo Instituto Brasileiro de Geografia e Estatística (IBGE), "investigou os produtos relacionados ao setor mais moderno da economia, ou seja, os serviços de informação, que abrangem as atividades relacionadas às novas tecnologias de comunicação e informação, os produtos das atividades tradicionais de transportes e parte dos serviços qualificados prestados às empresas, abrangendo serviços de engenharia e arquitetura" (BRASIL, 2006, p.1). A figura 1.3 apresenta um gráfico comparativo entre os serviços envolvidos nos estudos do PAS 2006 e mostra que o setor de serviços da informação foi um dos mais representativos no que diz respeito a sua "Receita

Operacional Líquida", que supera a R$ 143 bilhões, ficando atrás somente dos serviços de transporte que alcançou R$ 149 bilhões.

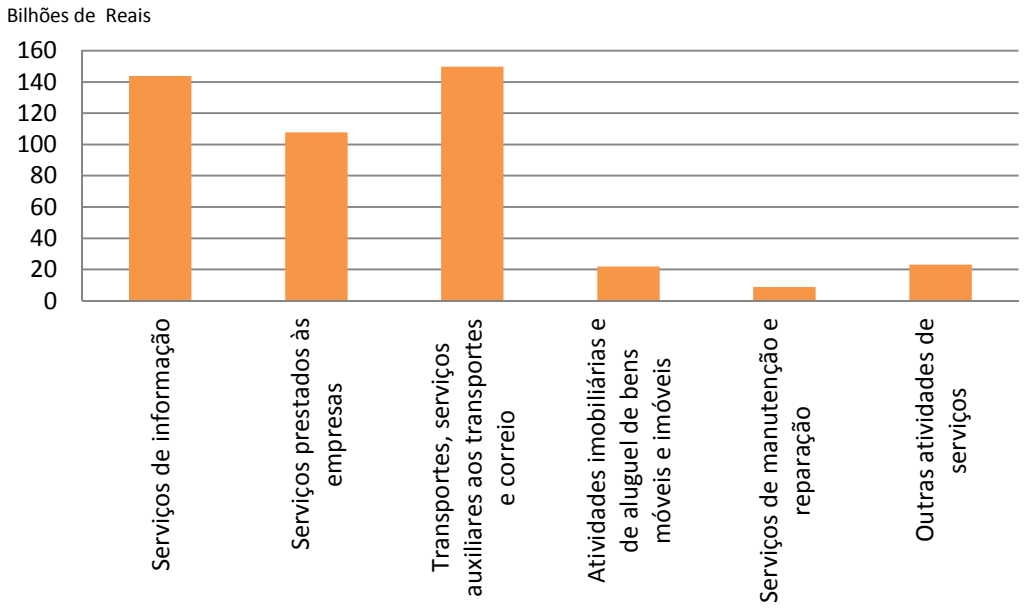

Figura 1.3 - Receita Operacional Líquida por atividades dos Serviços.
Fonte: IBGE, Pesquisa Anual de Serviços 2006.

Junto à emergência e consolidação das TIC surgiu um grande desafio: proporcionar aos sistemas educacionais a efetiva apropriação dessa realidade, a fim de se criarem novos ambientes de aprendizagem, novas maneiras de se construir o conhecimento e, fundamentalmente, uma readequação dos papéis de docentes e discentes. É nessa perspectiva de mudança e inovação, que surgiu o uso da "Informática na Educação", ou seja: a inserção do computador no processo de ensino-aprendizagem de todos os níveis e modalidades educacionais. Nesse contexto, fez-se imperiosa uma melhor compreensão do uso do computador nos processos educacionais, buscando identificar sua validade enquanto ferramenta de aprendizagem e as implicações de seu uso por parte de professores e alunos.

1.1 – Educação em Nível Superior: a reação às demandas emergentes

A adoção das TIC por todos os segmentos sociais suscitou a criação de cursos – em níveis técnicos, superior e mesmo os livres – para preparar os profissionais

que exercem atividades específicas, tais como: programadores de computadores, técnicos em *hardware*, analistas de sistemas, administradores de banco de dados, engenheiros de *software* e tecnólogos em geral. No Ensino Superior destaca-se o surgimento dos cursos de Ciência da Computação, Engenharia da Computação, Licenciatura em Computação, Sistemas de Informação, Tecnologia de Redes e Banco de Dados, entre outros. Além disso, mesmo os cursos superiores das demais áreas do conhecimento vêm se adequando a essa necessidade e, gradativamente, incorporando unidades curriculares relacionadas às TIC em sua estrutura curricular.

A tabela 1.1 demonstra a evolução da criação dos cursos superiores relacionados às TIC no Brasil, especificamente os da área da computação.

Tabela 1.1 – Criação de cursos superiores relacionados à área de computação.

Ano de início de funcionamento	Ciência da Comp.	Engenharia da Comp.	Licenciatura em Comp.	Outros	Seqüenciais de FE	Sistemas de Informação	Tecnólogo	Total
1969	1	0	0	1	0	0	0	2
1972	1	0	0	0	0	0	0	1
1973	1	0	0	1	0	0	0	2
1974	1	0	0	1	0	0	2	4
1975	2	0	0	0	0	0	1	3
1976	0	0	0	0	0	0	1	1
1977	2	0	0	1	0	0	0	3
1980	1	0	0	0	0	0	2	3
1981	1	0	0	0	0	0	0	1
1983	2	0	0	0	0	0	1	3
1984	5	0	0	0	0	0	0	5
1985	5	0	0	2	0	0	3	10
1986	6	0	0	3	0	0	4	13
1987	7	1	0	2	0	1	1	12
1988	9	0	0	2	0	2	11	24
1989	4	1	0	0	0	4	10	19
1990	9	3	0	0	0	2	16	30
1991	6	0	0	2	0	0	0	8
1992	10	2	0	2	0	2	8	24
1993	6	1	0	3	0	1	4	15
1994	9	1	0	6	0	2	9	27
1995	9	1	0	2	0	11	19	42
1996	10	4	0	3	0	8	3	28
1997	12	4	1	4	0	6	2	29
1998	32	8	0	9	0	11	6	66
1999	36	8	4	5	4	34	10	101
2000	20	6	3	3	9	52	11	104
2001	22	9	24	6	14	96	31	202
2002	16	7	12	3	8	91	58	195
2003	14	4	7	4	7	59	99	194
2004	11	5	10	3	7	32	139	207
2005	9	5	4	9	1	50	133	211
2006	26	7	5	5	2	33	97	175
Total	**305**	**77**	**70**	**82**	**52**	**497**	**681**	**1764**

Fonte: Estatísticas da Educação Superior da área de Computação (SBC).

É necessário observar-se os dados acumulados ao final do período abordado (2006), quando os cursos relacionados se encontram distribuídos conforme os percentuais apresentados na figura 1.4.

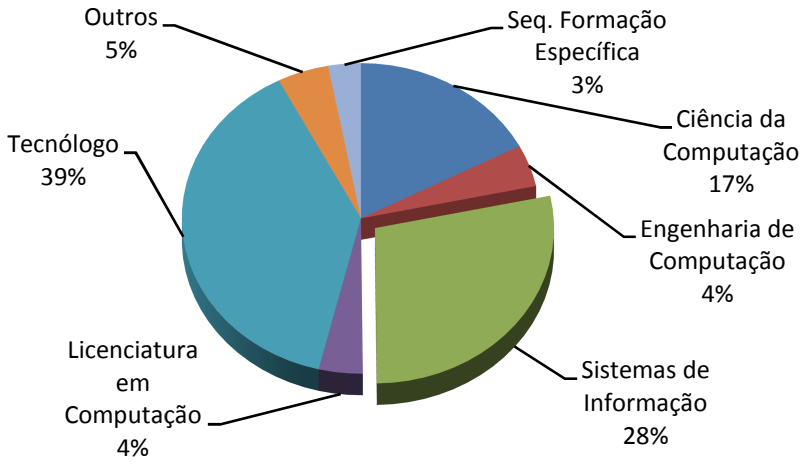

Figura 1.4 – Cursos da área de computação no final do ano de 2006.
Fonte: Estatísticas da Educação Superior da área de Computação

O termo "Tecnólogo" inclui os "Cursos Superiores de Tecnologia", tais como os de Banco de Dados e Tecnologia de Rede, que segundo o parecer CNE/CES 436/2001 do Ministério da Educação (MEC), se enquadram na sub-área "Informática" e exploram as "atividades de concepção, especificação, projeto, implementação, avaliação, suporte e manutenção de sistemas e de tecnologias de processamento e transmissão de dados e informações, incluindo *hardware*, *software*, aspectos organizacionais e humanos, visando a aplicações na produção de bens, serviços e conhecimentos" (Brasil, 2001, p.19). Esses cursos possuem uma duração mínima de 2000 horas e vêm tendo amplo crescimento. O curso de Sistemas de Informação, cenário deste trabalho, foi o que mais cresceu individualmente até o final de 2006.

O cenário atual de desenvolvimento de ferramentas tecnológicas para a educação está voltado predominantemente à Educação a Distância (EAD), que consiste no ensino por meio de mídia impressa ou eletrônica para pessoas engajadas em um processo de aprendizagem em tempo e local diferentes do(s) instrutor(es) e dos outros aprendizes (Lucena e Fucks, 2000). Inicialmente os processos dessa modalidade educacional eram efetivados pelo envio de material impresso via correio, o que foi ampliado com o advento

das tecnologias tradicionais como o rádio e televisão. Ultimamente, integrou-se o uso dos computadores e das redes telemáticas. O resultado dessa incorporação de recursos foi o redesenho da EAD, a qual passou a atender maiores contingentes populacionais, possibilitou maior interatividade e troca de experiências entre os sujeitos do processo, permitiu a melhor apresentação de materiais para estudo usando recursos de multimídia, entre outras.

Todas essas mudanças ainda estão em curso e geraram um fértil ambiente de pesquisas. Alguns indicativos apontam um caminho para a produção de conhecimentos nesse campo, por exemplo, Perrenoud (2000), propõe um deslocamento do foco de discussão da utilização ou não das TIC para a forma de sua utilização. Em comum acordo, Lévy (1999, p.26) afirma que "enquanto discutimos sobre os possíveis usos de uma dada tecnologia, algumas formas de usar já se impuseram". Quartiero (2007), ao discutir os trabalhos de Salomon (1992; 1998; 2001), amplia essa corrente de pensamento ao destacar que "[...] a principal preocupação que se deve ter quando se introduz uma nova tecnologia em sala de aula é com relação à qualidade da aprendizagem resultante do uso dessa tecnologia" (Quartiero, *op. cit.*, p.58).

A relevância desses aspectos torna-se ainda mais imperativa em ambientes educacionais onde o computador é um elemento indispensável, como é o caso do curso de Sistemas de Informação. Busca-se, portanto, construir um caminho para esse entendimento, levantando questões intrínsecas a processos educacionais próprios desses ambientes, avaliando as formas de uso dessas tecnologias e os desdobramentos de sua incorporação no fazer pedagógico.

1.2 – Estrutura de um curso de Sistemas de Informação

Como já foi dito, o curso de Sistemas de Informação (SI) foi eleito como ambiente educacional de estudo neste trabalho, que reúne em seu conteúdo programático elementos teóricos e práticos dos campos de tecnologia da computação e administração. O foco principal deste trabalho recorre à área de Tecnologia da Computação, a qual é estruturada em Fundamentos da Computação, Tecnologia da Computação e Sistemas de Informação, que apresentam, respectivamente, os seguintes conteúdos:

- Fundamentos da Computação: compreende o núcleo de unidades curriculares que envolvem os conceitos e as técnicas fundamentais relacionadas aos diversos ramos da computação;
- Tecnologia da Computação: compreende o núcleo de unidades curriculares que representam o conjunto de conhecimentos que possibilita a elaboração de soluções para problemas dos diversos domínios de aplicação;
- Sistemas de Informação: compreende o núcleo de unidades curriculares que visam à capacitação discente para a utilização dos recursos de Tecnologia de Informação para a solução de problemas nas atividades meio e fim de setores produtivos da sociedade.

As demais unidades curriculares do curso em questão estão organizadas dentro dos núcleos de Matemática e Contexto Social e Profissional (visão humanística das questões sociais e profissionais em consonância com a ética na computação). A figura 1.5 apresenta a distribuição de unidades curriculares por núcleo de acordo com a carga horária presente na estrutura curricular do curso em estudo.

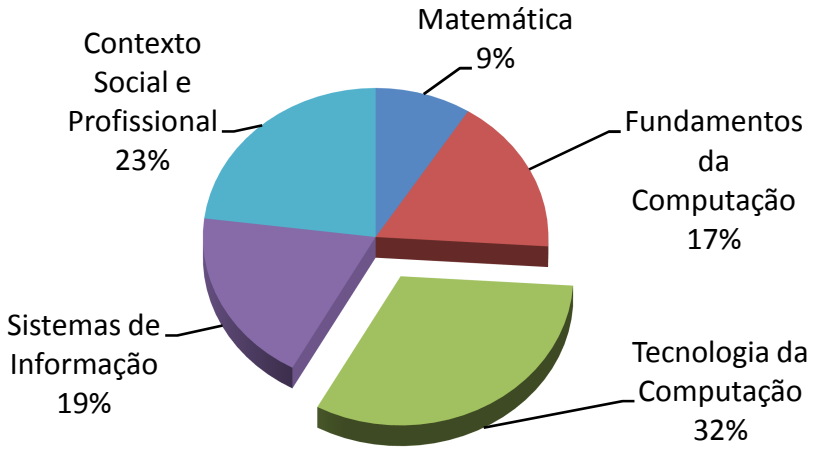

Figura 1.5 - Distribuição de unidades curriculares por núcleo de acordo com a carga horária no curso de SI.
Fonte: Plano de curso do Bacharelado em Sistemas de Informação da Universidade Beta

Nota-se que por ser um curso que tem ênfase no uso da tecnologia computacional, suas disciplinas estão concentradas nos núcleos de Tecnologia da Computação, Sistemas de Informação e Fundamentos de Computação, que, juntas, correspondem a 68% do curso. Destaca-se na figura 1.5 o núcleo de Tecnologia da Computação, que abriga as disciplinas de Linguagem de Programação, presentes em sete dos oito períodos do Curso.

CAPÍTULO II

COMPUTADORES NA EDUCAÇÃO: POSSIBILIDADES E CONCEITOS

A arte de aprender é uma órfã acadêmica.
Seymour Papert

No capítulo anterior evidenciou-se a emergência e a consolidação do uso das TIC pela sociedade contemporânea. Suscitou-se, embora brevemente, a apropriação dessas tecnologias pelo sistema educacional. Busca-se agora traçar um caminho que demonstre o uso do computador na educação. Logicamente, esse movimento de incorporação de uma nova tecnologia gera implicações para os sujeitos envolvidos no processo – professores e alunos – as quais se pretende, também, evidenciar.

Neste capítulo são introduzidos: o processo histórico do uso de computadores na Educação, as possibilidades de uso de computadores em ambientes educacionais (instrucionismo e construcionismo), os conceitos pertinentes à atividade cognitiva de programação de computadores bem como ao ferramental relacionado (linguagens de programação) e os subsídios teóricos do construcionismo. Compreende-se que esses referenciais são de fundamental importância para a melhor compreensão do processo de ensino-aprendizagem de programação de computadores.

2.1 – Máquinas de ensinar

Historicamente, a introdução dos computadores na educação tendeu a reproduzir o ensino através de máquinas. Em 1924, Sidney Pressey, arquitetou uma máquina para a correção de testes de múltipla escolha. Posteriormente, em 1950, Burrhus Frederic Skinner propôs sua máquina de ensinar, baseada na instrução programada (Souza e Fino, 2008). As máquinas de ensinar foram propostas por Skinner

como uma alternativa aos impasses que surgiram em decorrência das demandas de atendimento individual aos aprendizes.

> A instrução programada consiste em dividir o material a ser ensinado em pequenos segmentos logicamente encadeados e denominados módulos. Cada fato ou conceito é apresentado em módulos seqüenciais. Cada módulo termina com uma questão que o aluno deve responder preenchendo espaços em branco ou escolhendo a resposta certa entre diversas alternativas apresentadas. O estudante deve ler o fato ou conceito e é imediatamente questionado. Se a resposta está correta o aluno pode passar para o próximo módulo. Se a resposta é errada, a resposta certa pode ser fornecida pelo programa ou, o aluno é convidado a rever módulos anteriores ou, ainda, a realizar outros módulos, cujo objetivo é remediar o processo de ensino. (VALENTE, 1993, p.4)

Esse modelo de instrução foi bastante utilizado nas décadas de 50 e 60. Valente (1993) atenta para o fato de que de ele não prosperou dada a dificuldade de produção do material a ser utilizado e também à sua falta de padronização. O autor alerta para o surgimento do computador e, conseqüentemente, para a flexibilidade com que tal modelo passou a contar. Ainda que o uso de computadores fosse muito restrito e de elevado custo, empresas especializadas tais como a IBM e a RCA passaram a investir na produção de *softwares* que inauguram a instrução auxiliada por computador, ou *Computer Aided Instruction* (CAI) ainda na década de 60.

Com o advento dos microcomputadores na década de 80, o *software* CAI ganhou força, o que representou o início do processo de inserção dos computadores nas escolas, principalmente nos países desenvolvidos. Diversificaram-se, então, os tipos de *softwares* disponíveis. Além dos tutoriais, surgiram os programas de demonstração, exercício/prática, jogos didáticos e simuladores. Este fato esteve relacionado à investida em um processo de busca da eficácia no ensino. Prado (1999, p.19) amplia essa visão ao afirmar que o "computador, inserido nesse contexto, pode facilmente ser identificado e/ou incorporado como mais um instrumento que vem reforçar a ação educativa, centrada na eficiência das técnicas e dos métodos de ensino".

No CAI tem-se a primeira situação do uso do computador no contexto educacional como de uma máquina de ensinar aprimorada. Papert (1993/2008, p.52), ao referir-se a esse modelo de ensino, afirma que CAI consiste "em programar um computador para ministrar os tipos de exercícios tradicionalmente aplicados por um

professor em um quadro-verde, em um livro didático ou em uma folha de exercícios". Percebe-se que o computador, nesse processo, configurou uma nova roupagem aos artefatos tradicionais para a transmissão/replicação de conhecimento. Em outras palavras, "o uso do computador como máquina de ensinar consiste na informatização dos métodos de ensino tradicionais" (Valente, 1993b, p.32).

Em meio a esse panorama de inovação tecnológica, os sistemas educacionais passaram, mesmo que de forma discreta, a conviver com os computadores. Dentre os céticos que acompanharam esse evoluir, a figura de Seymour Papert destacou-se no questionamento de que qual seria a melhor via de integração dos computadores na educação. Papert é Sul Africano e tem formação em matemática. Dedicou-se a pesquisas na área de matemática na Cambridge University no período de 1954 a 1958. Posteriormente, transferiu-se para a Universidade de Genebra onde trabalhou de 1958 a 1963. No início da década de 60 filiou-se ao *Massachusetts Institute of Technology* (MIT). É um dos fundadores do MIT Media Lab e integrante do projeto "Um computador por criança", ao qual o governo brasileiro aderiu em 2005. Esse projeto prevê a disponibilização de um *Laptop* para cada criança em idade escolar, bem como sua utilização em sala de aula e em casa. O projeto está em andamento no País desde 25 de Janeiro de 2007, quando aconteceu o lançamento do projeto piloto no Rio Grande do Sul. A produção acadêmica de Papert relaciona-se com as áreas da Educação, Inteligência Artificial e Matemática.

Ao associar o uso dos computadores à Educação, Papert assumiu uma postura de "rebelião construtiva". Em sua concepção os computadores podiam e deviam ser utilizados "como instrumentos para trabalhar e pensar, como meios de realizar projetos, como fonte de conceitos para pensar novas idéias" (Papert, 1994, p.158) e não apenas como uma forma de apoio à instrução automatizada. Nesse sentido, Papert e sua equipe do *Massachusetts Institute of Technology* (MIT), entre os anos de 1967 e 1968, passaram a desenvolver uma forma de uso do computador que viabilizasse tais idéias: a ferramenta educacional LOGO.

Assim, passaram a coexistir no cenário educacional duas tendências relacionadas ao uso do computador, cada qual com características peculiares, que serão enfocadas nas seções seguintes deste capítulo.

2.2 – Instrucionismo: o computador como uma máquina de ensinar

Como visto anteriormente, a modalidade de CAI enfoca o uso do computador como uma máquina de ensinar e isso incorre, basicamente, em uma forma de replicar os métodos tradicionais de ensino, distinguindo apenas a forma de transmitir os conteúdos, que passam a ser mediados pelo computador. Essa abordagem pedagógica é tratada por Valente (1993) como instrucionismo, ou seja: o computador no processo educacional funciona como um suporte, reforço ou complementação ao que acontece na sala de aula.

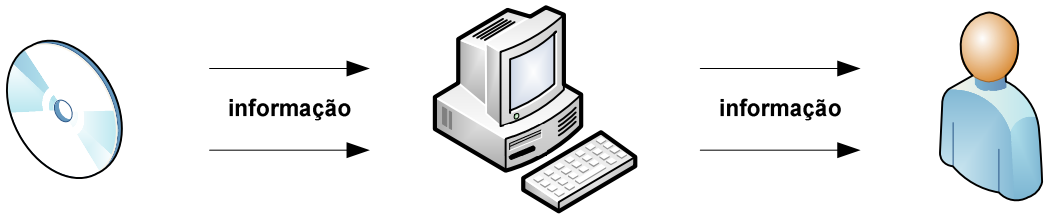

Figura 2.1 – Abordagem instrucionista de ensino.

A figura 2.1 sintetiza o processo instrucionista de ensino. Em um primeiro momento, o computador é provido das informações que serão ministradas ao aluno. Essa ação de municiar o computador com as atividades programadas para o ensino é realizada por meio da instalação de um *software* do tipo CAI. O processo de transmissão de conteúdos programados se perpetua quando um aluno faz uso do computador e, através dele, recebe o "pacote de informações" previamente programado. O aluno é o espectador para um volume de conhecimentos pré-determinados, pois, na maioria dos *softwares* de CAI, a interação existente entre o discente e o computador limita-se ao fornecimento de respostas a exercícios e a avanços ou retrocessos no conteúdo.

Dentro dessa abordagem enquadram-se os *softwares* de tutoriais, exercício e prática, jogos educacionais e os simuladores (Valente, 1993). Tutoriais são *softwares* que reproduzem a instrução programada, ou seja: "ensinam" um determinado conteúdo para o aluno. Geralmente são visualmente atrativos, possuem animações, som e texto usando o formato multimídia. *Softwares* de exercício e prática permitem ao educando a prática e revisão de conteúdos vistos em sala de aula. Usualmente envolvem um processo de memorização e repetição, apresentando questões de um dado assunto e, após a

apreciação/resposta do aluno, fornecem a solução da questão proposta. Os Jogos Educacionais são *softwares* que buscam combinar o atrativo ambiente de entretenimento oferecido pelos jogos convencionais (não pedagógicos), com a possibilidade de o aluno explorar algum conteúdo escolar específico. Como em todo "passa-tempo", existem as regras e também um objetivo específico a ser alcançado para vencer o jogo. Os Simuladores: são *softwares* que provêem um ambiente virtual onde o aluno pode moldar e explorar diferentes situações, por exemplo: estruturar circuitos elétricos sem correr riscos. Além disso, os simuladores permitem também uma significativa economia com a compra de equipamentos dispendiosos (ver figura 2.2).

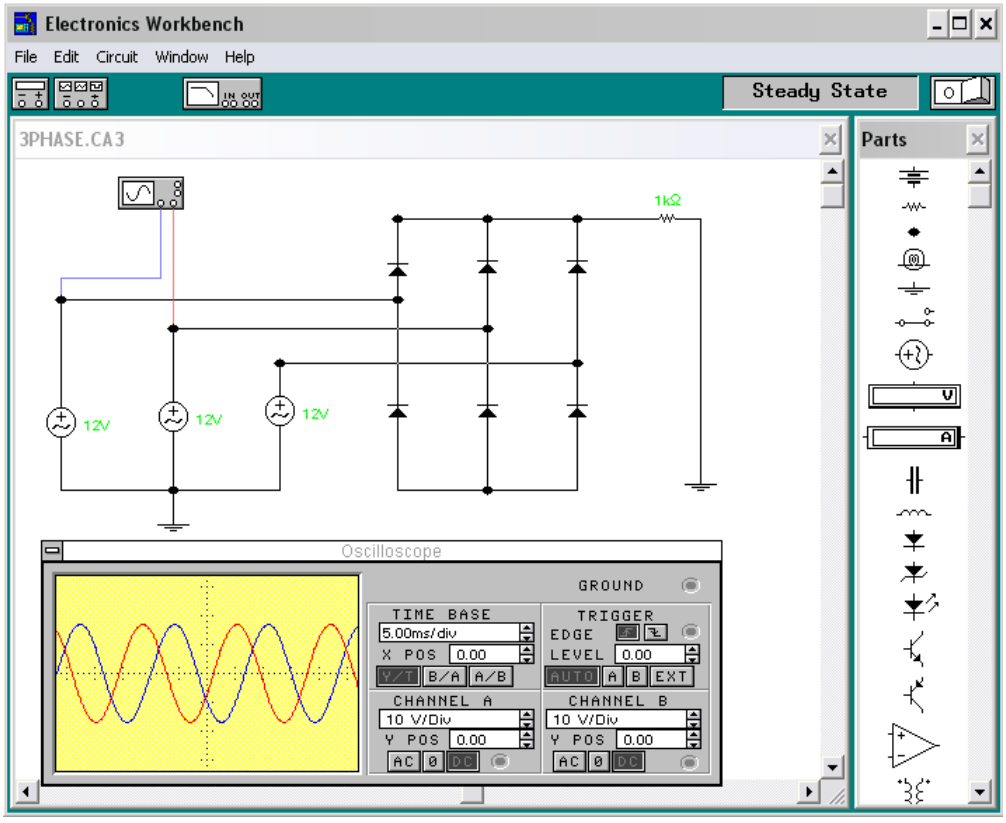

Figura 2.2 – Tela do programa *Eletronic Workbench* simulando um circuito trifásico e a leitura de duas dessas em um osciloscópio.
Fonte: Captura da tela do *software.*

A abordagem instrucionista teve e continua a ter espaço dentro do cenário da informática na educação. Foi a partir dela que os computadores começaram a ser difundidos nos ambientes escolares, sendo isso um ponto de partida para a criação de reflexões e novas possibilidades. Uma delas é que o uso do computador em um ambiente

de aprendizagem pode e precisa extrapolar a automatização da transmissão de conteúdos programáticos. É dentro dessa perspectiva que surge o computador como uma ferramenta educacional, tal como idealizada por Seymour Papert.

2.3 – Construcionismo: o computador como ferramenta de aprendizagem

Assumindo um panorama alternativo à linha instrucionista, surgiu o computador como uma ferramenta educacional. Valente (1993, p.12) explica que "segundo esta modalidade, o computador não é mais o instrumento que ensina o aprendiz, mas a ferramenta com a qual o aluno desenvolve algo, e, portanto, a aprendizagem ocorre pelo fato de estar executando uma tarefa por meio do computador". Fica explícita a idéia de que com o "computador ferramenta" o aluno será o sujeito promotor de uma ação, ou seja: seu lugar deixa de ser o de espectador e passa a ser o de agente.

Existem vários *softwares* que podem propiciar o uso do computador como uma ferramenta, com este sentido estrito, tal como tratado aqui. Entre eles destacam-se as planilhas eletrônicas, os gerenciadores de bancos de dados, os mecanismos de busca na internet, as ferramentas de cooperação e comunicação em rede e também as linguagens de programação.

Planilhas eletrônicas são *softwares* que permitem a criação e manipulação de folhas de cálculo, gráficos e também armazenar informações visando pesquisa, relatórios e estatísticas. São exemplos de planilhas o Excel do pacote de aplicativos *Microsoft Office* e *Calc* do pacote de aplicativos *OpenOffice*.

Os Gerenciadores de bancos de dados permitem criar coleções de informações em um formato devidamente estruturado, de forma a proceder a sua rápida recuperação (pesquisa), relacionamento e compartilhamento. Os gerenciadores de bancos de dados constituem-se como base para os sistemas de informações que atendem a diversas áreas, sendo largamente usados, por exemplo, em bibliotecas, hospitais, comércio, indústria, internet etc. Exemplos: *MySql*, *Oracle*, *Firebird*, *Postgress* e *MS-SQL Server*.

Os mecanismos de buscas na internet são ferramentas que permitem ao usuário realizar buscas a conteúdos específicos dentro da rede mundial de computadores.

Ultimamente tem sido o ponto de partida para a navegação na rede, sendo muito conhecidos o *Google, Alta Vista, Yahoo*, entre outros.

As ferramentas de cooperação e comunicação em rede constituem meios virtuais de troca de mensagens e ações cooperativas na Internet. Enquadram-se o correio eletrônico (*e-mail*), as ferramentas de troca sincrônica de mensagens (*MSN*, por exemplo) e também as plataformas de EAD tais como o *Moodle* e o Teleduc.

Linguagens de programação são *softwares* que proporcionam um ambiente de expressão de raciocínio visando à solução de problemas por meio do computador. Em outras palavras, uma linguagem de programação permite a criação de *softwares* específicos. *BASIC, PASCAL, FORTRAN, C++, JAVA, LOGO* são nomes de algumas dessas ferramentas.

Ainda no prefácio de seu livro "LOGO: computadores e Educação" (tradução do original *Mindstorms – Children, Computers and Powerfull Ideas)*, Papert ao criticar o paradigma instrucionista, introduz o seu pensamento mostrando que o computador pode e deve ser utilizado como uma máquina de produção de conhecimento.

> [...] a frase "instrução ajudada pelo computador" (*computer-aided-instruction*) significa fazer com que o computador ensine a criança. Pode-se dizer que o computador está sendo usado para "programar" a criança. Na minha perspectiva é a criança que deve programar o computador e, ao fazê-lo, ela adquire um sentimento de domínio sobre um dos mais modernos e poderosos equipamentos tecnológicos e estabelece um contato íntimo com algumas das idéias mais profundas da ciência, da matemática e da arte de construir modelos intelectuais. (PAPERT, 1980/1985, p.17)

Nessa forma alternativa de uso da máquina, alunos e professores passam a ter a chance de elaborar projetos para solução de situações-problemas das mais diversas áreas. Isso pode ser conseguido, por exemplo, com o uso das linguagens de programação como suporte à elaboração de programas de computador que representam essas soluções.

Papert (1986) sugeriu o termo construcionismo para designar a modalidade em que um aluno utiliza o computador como uma ferramenta com a qual ele constrói seu conhecimento. Valente (1993) afirma que Papert usou o termo

construcionismo para “mostrar um outro nível de construção do conhecimento: a construção do conhecimento que acontece quando o aluno elabora um objeto de seu interesse, como uma obra de arte, um relato de experiência ou um programa de computador” (Valente, *op. cit.*, p.40). Percebe-se que o uso do computador nessa abordagem se configura de maneira antagônica à inicialmente introduzida com o instrucionismo.

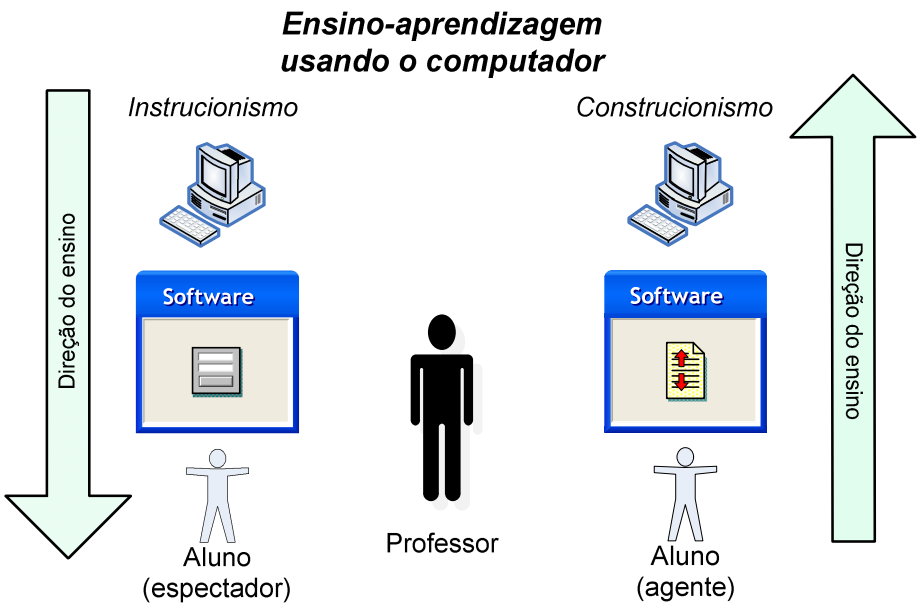

Figura 2.3 – Linhas de ensino-aprendizagem usando o computador.

A figura 2.3 apresenta as duas linhas de ensino-aprendizagem – instrucionismo e construcionismo – sugerindo uma comparação entre elas. Nota-se, em ambos os casos, a presença do computador, do aluno, de um professor e de um *software*. As diferenças estão no sentido da direção do ensino, no tipo de *software* utilizado, na postura a ser adotada pelo educador e na caracterização do aluno dentro do processo. Como visto anteriormente, no instrucionismo tem-se o computador pré-programado ensinando a um aluno – espectador do processo – por meio de um *software* da modalidade CAI. No construcionismo de Papert o processo é invertido. O educando precisa assumir postura ativa e passar a ensinar ao computador a cumprir uma determinada tarefa. Isso é conseguido por meio de um *software*, que em nosso caso é uma LPC. Configura-se nas duas abordagens a mediação do processo de aprendizagem pelo professor. Por hora, indica-se que a atuação desse tutor deverá ser compatível com

as possibilidades oferecidas por cada linha. Em tempo, serão apresentadas as características inerentes a esses profissionais, as quais são decisivas para o sucesso de suas intervenções pedagógicas.

A abordagem construcionista é sintetizada em seu objeto de estudo: um problema e a sua compreensão, a elaboração de uma estratégia de solução no computador, pelo aluno, mediado por um profissional da educação; e no ferramental: um computador e uma linguagem de programação usados para a construção do conhecimento. Baseando-se nisto, a próxima seção enfoca o conceito de linguagem de programação e seu uso no meio educacional.

2.4 – Linguagens de programação: meio de expressão de idéias

De acordo com Ascencio (1999), o propósito do uso de um computador está ligado a sua versatilidade, capacidade de processamento e segurança ao processar dados, ou seja: receber dados em um dispositivo de entrada (teclado, *mouse*, *scanner* etc.), realizar transformações nesses dados e fornecer uma resposta em um dispositivo de saída (monitor de vídeo, impressora etc.). De posse da saída de informações, essas podem ser melhoradas ou ainda, corrigidas. Isto constitui uma etapa de realimentação das entradas de dados num processo conhecido como *feedback* (retroalimentação). Estas etapas são ilustradas na figura 2.4.

Figura 2.4 – Etapas do processamento de dados.

Para proceder às etapas do processamento de dados, o computador faz uso de suas partes físicas, também conhecidas como *hardware*, que para se tornarem operacionais necessitam, sobretudo, de *softwares* (programas). Para a criação desses programas são usadas as linguagens de programação de computadores (LPC). “Uma linguagem de programação assemelha-se a uma língua natural, humana, na medida em que favorece certas metáforas, imagens e maneiras de se pensar” (Papert, 1980/1985 p.

52). As LPC também são *softwares,* porém, possuidores de um vocabulário próprio, sintático[2] e semântico[3], não ambíguo que propicia a determinação de instruções para um computador.

> As linguagens de programação constituem-se em uma ferramenta de concretização de produto de *software*, que representa o resultado da aplicação de uma série de conhecimentos que transformam a especificação da solução de um problema em um programa de computador que efetivamente resolve aquele problema. (SANTOS E COSTA, 2006, p. 41)

Existem inúmeras linguagens de programação, cada qual com características peculiares e propósitos diferentes. Destaca-se por hora, a linguagem PASCAL, por estar sendo utilizada, freqüentemente, em cursos superiores que possuem as unidades curriculares de programação (Timóteo e Brasileiro, 1998).

A linguagem PASCAL foi projetada para o ensino de programação. Ela foi desenvolvida em 1968 pelo professor Niklaus Wirth, do Instituto de Informática da ETH (*Eidgenössische Technische Hochschule*), em Zurique, Suíça. Sua denominação foi uma homenagem ao matemático Blaise Pascal, inventor da primeira máquina de calcular. O desejo de seu criador era dispor de uma ferramenta que fosse simples, coerente e, ao mesmo tempo, que incentivasse a confecção de programas de computador (Farrer *et al*, 1995). Em 1973, foi adotada academicamente pela Universidade da Califórnia, em San Diego (EUA) e uma década mais tarde, a soft-house norte-americana *Borland Internacional* lançou o Turbo Pascal (figura 2.5), o que consolidou sua notoriedade (Machado, 2005).

[2] Do ponto de vista sintático, uma linguagem de programação possui notações que podem ser utilizadas para especificar ações a serem executadas por um computador.

[3] Do ponto de vista semântico, uma linguagem de programação compreende um conjunto de conceitos que um programador usa para resolver problemas.

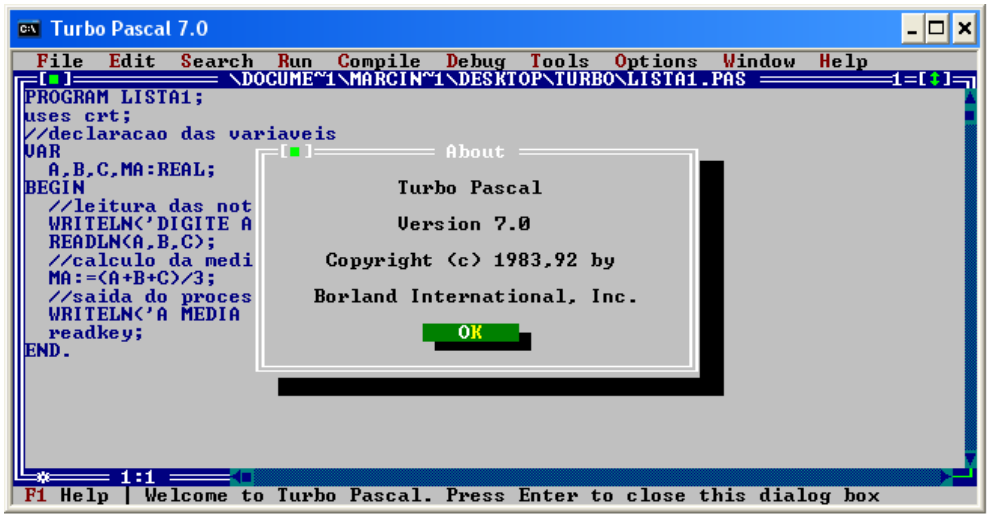

Figura 2.5 - Ambiente de programação do Turbo Pascal, versão 7, da Borland Internacional.

Um fato que confirma a popularidade e a utilização da linguagem Pascal para fins educativos é a existência de ferramentas livres regidas pela Licença Pública Geral (GNU *General Public Licence)*. Esse licenciamento assegura aos usuários a liberdade de executarem, copiarem, distribuírem, estudarem, modificarem e aperfeiçoarem o *software*. Além disso, os utilizadores estão desobrigados de pagarem qualquer tipo de taxa e possuem a garantia de que esses produtos não se tornarão proprietários (como o Turbo Pascal da *Borland International, Inc*.). Uma dessas ferramentas é o *BloodShed Dev-Pas*, exibida na Figura 2.6.

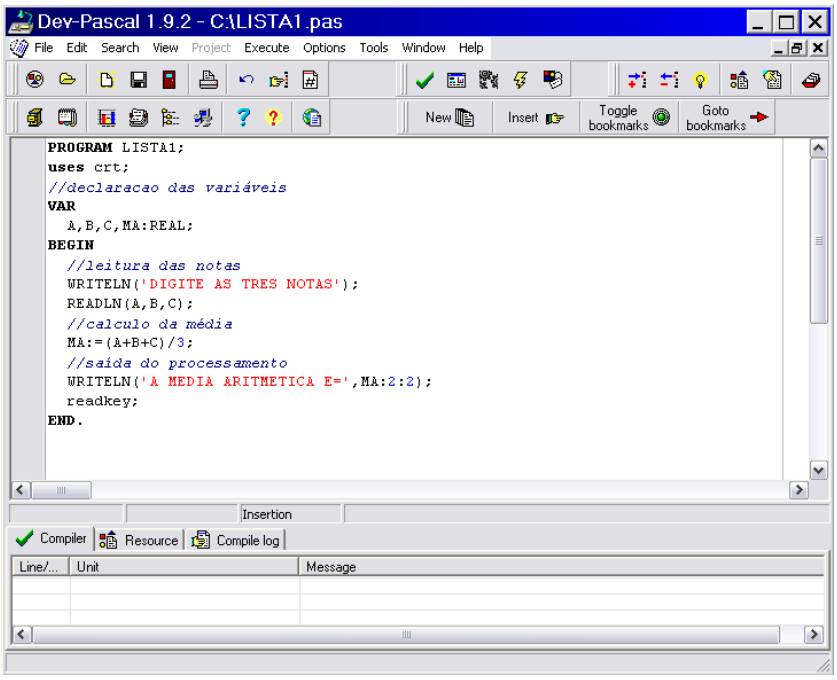

Figura 2.6 – A ferramenta BloodShed Dev-Pas, regida pela GNU.

O uso das LPC na educação está presente em diferentes níveis e domínios do conhecimento. Dessa forma, é muito comum vermos cursos como os de matemática ou engenharias, entre outros, possuírem unidades curriculares de introdução à programação de computadores, embora, logicamente, sejam os cursos ligados à área da computação os que possuem uma maior carga horária dedicada à exploração de tais ferramentas. Geralmente, essas unidades curriculares exploram a atividade de resolução de problemas por intermédio das LPC. Valente (1993) esclarece que:

> As linguagens para representação da solução do problema podem, em princípio, ser qualquer linguagem de computação, como o BASIC, o Pascal, ou o Logo. No entanto, deve ser notado que o objetivo não é ensinar programação de computadores e sim como representar a solução de um problema segundo uma linguagem computacional. O produto final pode ser o mesmo – obtenção de um programa de computador – os meios são diferentes. Assim, como meio de representação, o processo de aquisição da linguagem de computação deve ser a mais transparente e a menos problemática possível. Ela é um veículo para expressão de uma idéia e não o objeto de estudo. (VALENTE, 1993, p.14)

Percebe-se, portanto, que o objetivo do uso das LPC é propiciar um ambiente para a construção de programas de computador. Elas representam uma via de expressão de idéias, sendo essas últimas o foco do processo educativo. A atividade cognitiva de construção de programas é dita programação de computadores. Assim, para Papert, “programar significa, nada mais, nada menos, comunicar-se com o computador, numa linguagem que tanto ele quanto o homem podem ‘entender’” (Papert, 1980/1985, p.18). Nesse sentido, o uso das LPC no computador caracteriza-o como uma ferramenta, pois elas propiciam a representação das soluções de problemas, expressas por seqüências lógicas de ações. Isso equivale a dizer que o aluno está tutelando, ou ainda, ensinando o computador a resolver uma tarefa no formato de um programa (Almeida, 1999).

> Elaborar um programa significa manipular um sistema de palavras e de regras formais, que constituem a sintaxe e a estrutura da linguagem, que dão suporte para se representar os conhecimentos e as estratégias necessários à solução do problema. O conhecimento não é fornecido ao aluno para que ele dê as respostas. É o aluno que coloca o conhecimento no computador e indica as operações que devem ser executadas para produzir as respostas desejadas. (ALMEIDA, *op. cit.*, p.19)

A elaboração da solução de uma situação-problema pode ser expressa, inicialmente, através de algoritmos. Forbellone e Eberspächer (1999) introduzem o conceito de algoritmo como sendo uma seqüência de passos que visa atingir um objetivo bem definido. Em comum acordo, tem-se em Ascencio (1999) o conceito de algoritmo como uma seqüência de ações que deve ser seguida para a realização de uma tarefa. Maltempi e Valente (2000) associam a atividade de programação com o desenvolvimento dos algoritmos e destacam que eles constituem a parte construtiva da atividade.

Como exemplo ilustrativo, apresenta-se um possível algoritmo que propõe uma solução de uma situação do tipo: "Como ensinar a um computador a somar dois números inteiros quaisquer fornecidos por um usuário?"

Ascencio e Campos (2002) sinalizam que um algoritmo pode ser construído por intermédio de uma discrição narrativa, ou seja, a expressão de um raciocínio em uma linguagem como a língua portuguesa. Inicialmente, pode ser proposta a seguinte seqüência de passos: PASSO 1 – Pedir os números para a soma; PASSO 2 – Ler esses números; PASSO 3 – Processar a soma dos dois números lidos anteriormente no passo 2; PASSO 4 – Apresentar o resultado. Essa construção, entretanto, pode evoluir para a forma de pseudocódigo, conforme exemplificado a seguir:

ALGORITMO Soma_de_dois_números_quaisquer;
DECLARE a, b, soma: NUMERICO INTEIRO[4];

1. ESCREVA "Por favor, digite dois números inteiros para a realização da soma";
2. LEIA a,b;
3. soma = a+b;
4. ESCREVA "A soma dos números digitados é igual a ", soma;

FIM ALGORITMO.

Essa elaboração mental, geralmente, é feita em papel. Entretanto, pode ser descrita, executada, depurada[5] e em um computador por intermédio das linguagens

[4] A expressão "numérico inteiro" indica que "a, b e soma" são números inteiros.

de programação. Reitera-se que elas propiciam um ambiente adequado à transcrição dos algoritmos utilizando um vocabulário próprio e acabam por estruturar o raciocínio construído sob a forma de programas de computador, o que pode ser observado na Figura 2.7.

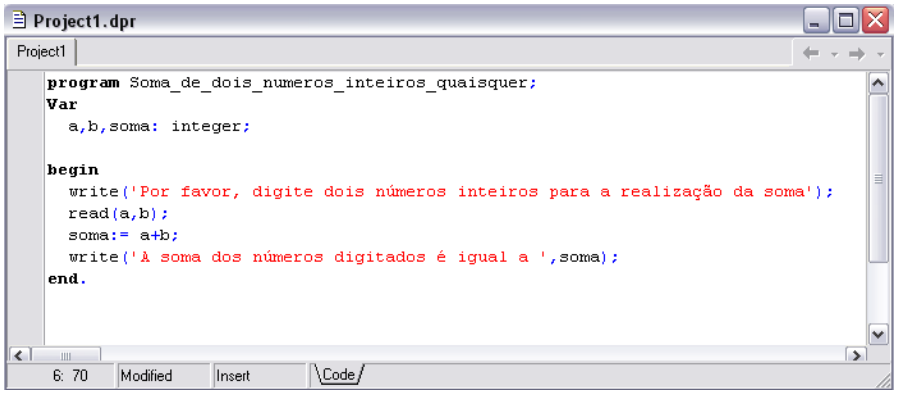

Figura 2.7 – Algoritmo escrito sob a forma de programa usando PASCAL.

A atividade de programação exige o domínio de uma LPC por parte do programador, o que irá possibilitar a codificação do programa e seu processamento pelo computador. Ela também requer o conhecimento da situação-problema abordada e alguma criatividade, uma vez que uma solução pode ser expressa de diferentes maneiras. A construção de programas inibe, portanto, a reprodução e memorização de informações e requer a formalização de raciocínio lógico, reflexão, dedicação à atividade e pesquisa em relação ao problema a ser modelado e também quanto à LPC adotada (Maltempi e Valente, 2000). Seymour Papert, ainda na década de 60, vislumbrou todas essas possibilidades – desejáveis a um ambiente de aprendizagem – e, buscando explorá-las, investiu na elaboração de sua teoria construcionista.

A seção seguinte dedica-se ao detalhamento da atividade de programação de computadores, mostrando que o seu resultado é significativo do ponto de vista da construção de conhecimento, constituindo-se como prática base da proposta construcionista de Papert.

[5] O processo de descrição, execução e depuração será discutido nas seções seguintes desse trabalho e constitui a base da atividade cognitiva de programação de computadores.

2.5 – A atividade de programação de computadores

A afirmativa de que os ambientes educacionais são pobres em recursos que estimulem o pensamento e a expressão de idéias é habitual. Parte desse pensamento encontra força no próprio sistema educacional, que preserva sua base de ensino apoiada na transmissão e reprodução de conhecimento, na repetição e memorização de informações, o que é caricaturado na figura 2.8. Papert (1993/2008) lembra que o educador brasileiro Paulo Freire criticava essas práticas, e recorda a metáfora sugerida por ele, onde a escola seguia um "modelo bancário" no qual pequenas porções de informações seriam depositadas na mente dos educandos, em conformidade ao que acontece com dinheiro em uma conta.

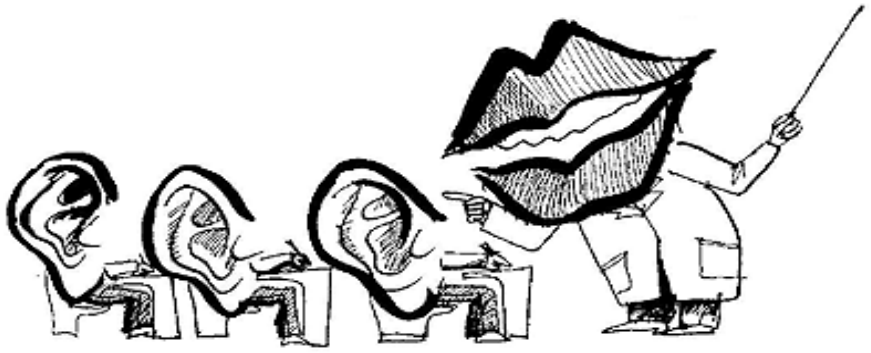

Figura 2.8 – Caricatura do método transmissivo de ensino.
Fonte: HARPER, Babette et al, 1980, p.48.

Para esse autor, portanto, o computador é uma ferramenta alternativa à demanda de um fazer educativo com bases mais sólidas, assim, "o acesso aos computadores pode mudar completamente essa situação" (Papert, 1980/1985, p.45). Isto significa que o computador pode oferecer a seus usuários a possibilidade de pensar, refletir, expandir-se e, o principal: implementar suas idéias. O meio idealizado para a concretização das proposições de Papert, baseia-se no uso de uma linguagem de programação visando à concretização de um processo de construção de conhecimento. Para isso Papert idealizou a linguagem de programação LOGO, base de sua proposta construcionista, a qual será explorada no capítulo III.

Como visto anteriormente, uma LPC propicia o exercício da programação. Essa atividade permite, fundamentalmente, representar a solução de um problema qualquer, sobre a forma de um programa de computador. Nesse sentido, "Elaborar um programa significa manipular um sistema de palavras e de regras formais, que constituem

a sintaxe e a estrutura da linguagem, que dão suporte para se representar os conhecimentos e as estratégias necessárias à solução do problema" (Almeida, 1999, p.19). Essa prática é iniciada pela existência de um problema para o qual se almeja uma solução. Do ponto de vista educacional, é a partir daí que se estabelece a interação sujeito aluno X computador X sujeito professor, na qual o sujeito aluno passa a descrever suas ações para serem executadas pelo computador. Na seqüência, o aluno pode refletir sobre suas próprias idéias e depurá-las dentro de um processo de retroalimentação. Essa última é provida pelo *feedback* do computador, ou mesmo pelas intervenções do professor e/ou colegas (Altoé e Penati, 2005).

Dada a relevância da interação estabelecida na atividade cognitiva de programação de computadores, vários autores – Valente (1993, 1999, 2002), Almeida (1999), Maltempi e Valente (2000), Altoé e Penati (2005), Freire e Prado (1995) – descrevem-na em quatro etapas, a saber: **descrição** da resolução do problema nos termos da linguagem de programação, **execução** dessa descrição pelo computador, **reflexão** sobre o que foi produzido pelo computador e **depuração** dos conhecimentos por intermédio da busca de novas informações ou do pensar. Este processo será tratado doravante pelo acrônimo "DERD".

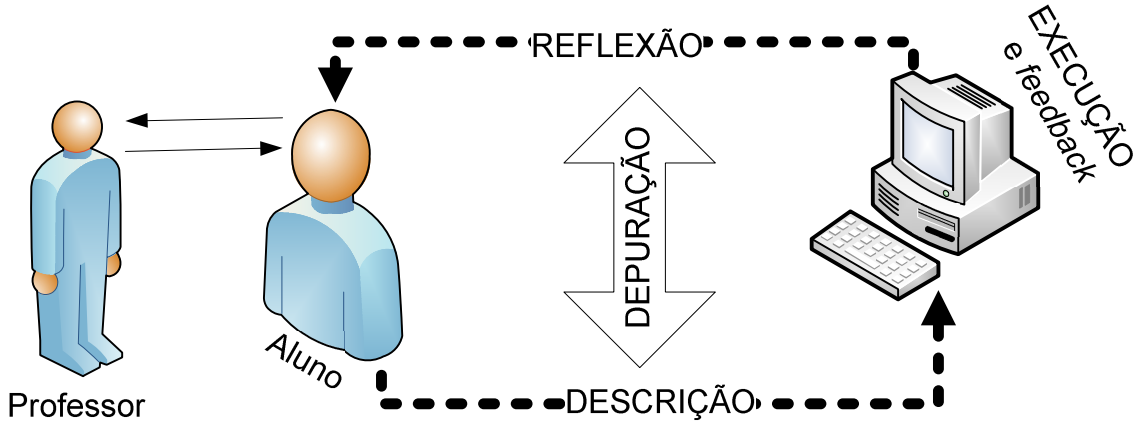

Figura 2.9– Interação aluno-computador-professor estabelecida na atividade de programação. Adaptado de Valente (1993)

A figura 2.9 ilustra os quatro estágios do processo DERD, conforme:

- ETAPA 1 – Descrição da resolução do problema em termos de linguagem de programação: Após a apreciação do problema a ser resolvido, o aluno usa sua estrutura de conhecimentos – conceitos relativos à questão, estratégias de aplicações dos conceitos, conceitos inerentes à linguagem de computador e também sobre o próprio computador – para explicitar, passo a passo, a solução do problema. Reitera-se que essa descrição vem ao encontro da elaboração de algoritmos sob a forma de pseudocódigos (seção 2.4), os quais sintetizam esse fazer. Em outras palavras, de posse de um algoritmo, um discente pode simplesmente transcrevê-lo através de um teclado (periférico de entrada – ver Figura 2.4) seguindo a sintaxe e a semântica da LPC (A figura 2.6).
- ETAPA 2 – Execução da descrição pelo computador: Uma vez implementada a descrição do problema usando a LPC, essa codificação pode ser lida, interpretada e executada pelo computador (processamento dos dados – ver Figura 2.4). Ao executar, o computador irá fornecer um *feedback* fiel e imediato ao educando, daquilo que foi solicitado à máquina.
- ETAPA 3 – Reflexão sobre o que foi produzido pelo computador: A máquina configura-se no processo como executora das tarefas solicitadas pelo aluno. Ao fazê-lo, fornece o resultado, geralmente usando o monitor de vídeo ou mesmo uma impressora (periférico de saída de informações – ver Figura 2.4). Nesse ponto do processo o aluno avalia, interpreta e reflete sobre o resultado fornecido pelo computador (*feedback*). A atividade pode provocar diferentes situações: o aluno alcança o sucesso, uma vez que sua descrição conseguiu suprir a solução do problema e finaliza a atividade; acontece um erro na descrição ou durante a execução (*bug*) – esse é apresentado ao aluno para correção; ou ainda,

mesmo que o aluno tenha conseguido uma solução satisfatória, ele quer melhorar ainda mais sua construção. Nas duas últimas situações mencionadas, o processo evolui para o estágio de depuração.

- ETAPA 4 – Depuração dos conhecimentos por intermédio da busca de novas informações ou do pensar: Na etapa 3, o computador pode acusar um erro de sintaxe (por exemplo, um termo da linguagem que foi escrito erroneamente), ou mesmo um equívoco na lógica empregada na construção do programa. No primeiro caso, o aluno terá de rever os conceitos da linguagem utilizada e proceder à correção do que está conflitante. No segundo caso, ele precisará reconsiderar sua estratégia de solução, buscando melhorá-la e adequá-la. Como visto, ainda na etapa 3, ao executar seu programa o aluno pode obter sucesso, mas ainda assim desejar melhorar algum detalhe de sua construção. Nesse ponto, ele passa também a rever suas estratégias e conceitos utilizados na sua representação. Para todos os casos aqui salientados, o processo de pós-depuração da solução inicial implica em uma nova descrição, execução, reflexão e depuração. É um processo contínuo, que se perpetua até que o educando se dê por satisfeito.

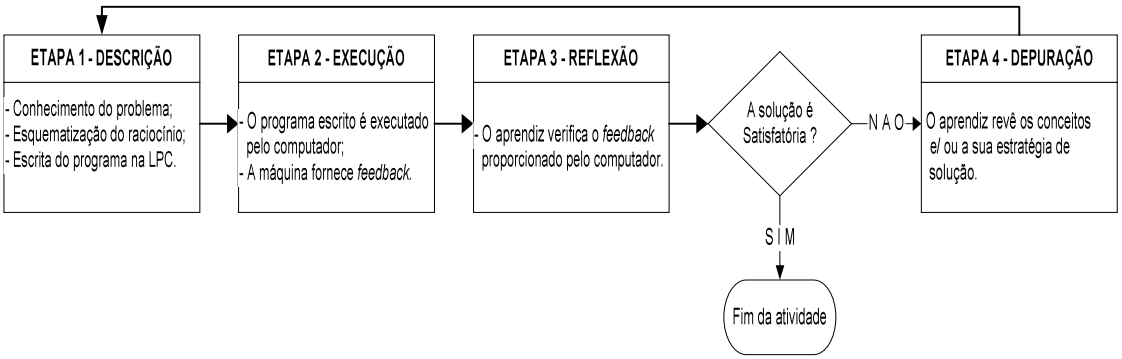

Figura 2.10 – Esquema do processo de DERD.

A Figura 2.10 sintetiza e esquematiza as quatro etapas do processo de DERD. Esses estágios são interdependentes. Vale lembrar, entretanto, que são os estágios de reflexão (etapa 3) e depuração (etapa 4), que quando bem explorados, possibilitam a concretização do conhecimento por parte do aluno. Isso acontece, essencialmente, quando o discente encontra uma situação não satisfatória relacionada a seu fazer. Papert (1980/1985, p. 39) afirma que "quando se aprende a programar um computador, dificilmente se acerta na primeira tentativa". Do ponto de vista construcionista isso não representa algo condenável, muito pelo contrário: o erro é tido como oportunidade ideal para a construção do conhecimento. Almeida (1999, p.23) destaca que "O erro passa a ser então um revisor de idéias e não mais um objeto de punição, intimidação e frustração". Da mesma forma, Valente (1999, p. 75) diz que "o processo de achar e corrigir um erro constitui uma oportunidade única para o aprendiz aprender sobre um determinado conceito envolvido na solução do problema ou sobre estratégias de resolução de problemas".

Nota-se, portanto, que após o aluno receber o *feedback* do computador sobre a execução de seu programa, ele passa a tentar identificar a origem de um erro e saná-lo, ou ainda, empenhar-se na construção de melhorias em seu programa. Esse processo de depuração é, para o aluno, um momento de "pensar sobre o pensar" (Turkle, 1984). A atividade de depuração na programação de computadores é valiosíssima do ponto de vista da aprendizagem e é exatamente por isso que deve ser estimulada. Assim, o processo de DERD, como um todo, configura-se como um exercício árido, que demanda esforço, dedicação, concentração e motivação por parte do aluno. Nesse ponto, torna-se imprescindível a atuação de um profissional da educação, que favoreça a aprendizagem do aluno (ver figura 2.9).

> [...] o professor precisa compreender a representação da solução do problema adotada pelo aluno; acompanhar a depuração e tentar identificar as hipóteses, os conceitos e os possíveis equívocos envolvidos no programa; e assumir o erro como uma defasagem ou discrepância entre o obtido e o pretendido. Assim, o professor intervém no processo de representação do aluno, ajuda-o a tomar consciência de suas dificuldades e a superá-las; a compreender os conceitos envolvidos; a buscar informações pertinentes; a construir novos conhecimentos; e a formalizar esses conhecimentos. (ALMEIDA, 1999, p. 23)

Assim, constitui-se como uma idéia errônea, o pensamento de que, nessa dinâmica, o discente aprende sozinho; que basta apresentar-lhe um problema, colocá-lo na frente de um computador para que o processo de DERD se consolide. Muito pelo contrário. O aluno pode, por exemplo, incorrer numa situação em que ele não sabe um conceito – o que representaria uma estagnação do processo. Daí a importância do suporte a ser suprido pelo agente educacional. Destaca-se, por hora, que além de elucidar e/ou sanar eventuais questionamentos do discente, o professor deve ser um estimulador do processo de DERD, mostrando-se disposto a cooperar e aprender em conjunto com seus alunos.

O processo de DERD foi inicialmente utilizado por autores – Valente (1993), Almeida (1999), Maltempi e Valente (2000), Altoé e Penati (2005) – como um ciclo de descrição-execução-reflexão-depuração. Entretanto, a idéia de ciclo remonta a uma perspectiva fechada, que tem pontos coincidentes para o início e o fim de cada iteração. Isso pode passar a idéia de que não existem incrementos no conhecimento do aluno. Em Valente (2002) essa posição é revista e o pesquisador anuncia que essa é uma idéia limitada. Propõe, então, que a construção de conhecimentos por intermédio da atividade de programação de computadores se aproxima de uma espiral.

> As ações podem ser cíclicas e repetitivas, mas a cada realização de um ciclo, as construções são sempre crescentes. Mesmo errando e não atingindo um resultado de sucesso, o aprendiz está obtendo informações que são úteis na construção do conhecimento. Na verdade, terminando um ciclo, o pensamento nunca é exatamente igual ao que se encontrava no início de sua realização. Assim a idéia mais adequada para explicar o processo mental dessa aprendizagem é a de uma espiral (VALENTE, 2002, p.27).

Essa nova perspectiva é reproduzida na figura 2.11, a qual mostra várias iterações de DERD configurando a aquisição de conhecimento em um processo crescente, no formato de uma espiral.

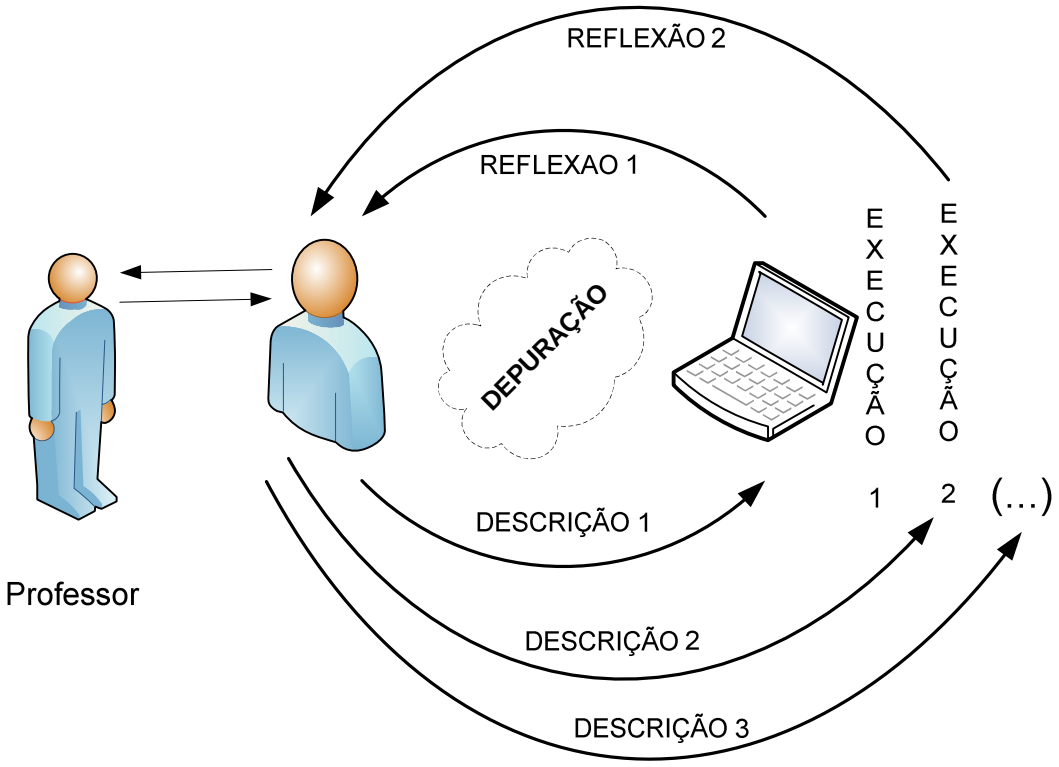

Figura 2.11 – A espiral do conhecimento seguindo o processo de DERD de programas de computador.

Essa espiral de conhecimento é descrita seguindo o processo de DERD a partir do momento em que um aluno se disponibiliza a construir um programa de computador visando a representar a solução de uma dada situação. Nomeia-se esse programa de “PRG”. Nesse momento, o aluno não precisa possui conhecimento completo das questões que envolvem a construção de “PRG” (conceitos sobre o problema, sobre a linguagem de programação e até mesmo sobre o computador). No início da atividade de programação, uma solução inicial “PRG1” é descrita usando a compreensão inicial do aluno. A execução “PRG1”, gera um resultado (*feedback*) “RST1”, o qual servirá de objeto da primeira reflexão sobre o programa, o que é denominado de “OBJ1”. Esse “OBJ1” pode acarretar um processo de depuração – “DPR1” – pelo qual será produzida uma nova versão para o programa inicialmente proposto – seguindo a linha de raciocínio: provavelmente, o “PRG2”. Fato é que “PRG2” incluirá um nível de conhecimentos mais sofisticados que na versão original “PRG1”, que advêm do processo de reflexão do discente, de pesquisas em livros e na internet, de conversas com membros de seu grupo de estudos e, não menos, da mediação da aprendizagem realizada por um professor. Na

busca da solução mais adequada do problema – “PRG” – o caminho percorrido em cada iteração no processo de DERD caracterizará um incremento nos conhecimentos dos sujeitos envolvidos (Valente, 2002).

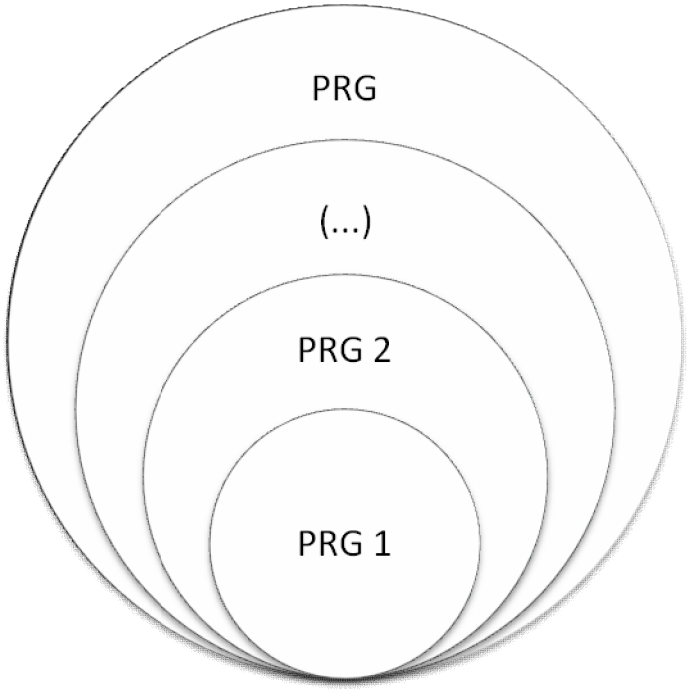

Figura 2.12 – Construção de um programa com a metáfora do crescimento da cebola.

É importante esclarecer que o processo de DERD é aplicável em qualquer linguagem de programação, sendo que cada uma dessas, possui uma estrutura peculiar que envolve formas diferentes para se representar a resolução de problemas. Na Computação, esse conceito é conhecido como paradigma de programação. Em Educação o paradigma mais freqüentemente explorado é o procedural (Almeida, 1999), também conhecido como imperativo, no qual “o computador é entendido como uma máquina que obedece ordens e o programa como uma prescrição da solução para o problema” (Baranauskas, 1993, p.46).

A seção seguinte apresenta sucintamente as idéias de alguns teóricos nas quais Papert se apoiou. Nesse sentido, adotou-se uma perspectiva otimista, pela qual foram considerados seus pontos complementares, os quais possibilitaram a concretização da abordagem construcionista.

2.6 – Subsídios teóricos do construcionismo

A implementação da proposta construcionista de Papert remonta à década de 60, quando o autor buscou – junto a sua equipe no MIT – elaborar um *software* educacional que possibilitasse o uso do computador como uma ferramenta de construção do conhecimento por parte de seus usuários. Esse *software* foi concebido no formato de uma linguagem de programação, batizada de LOGO, nome que referencia o termo grego que significa "pensamento, raciocínio e discurso", ou também, "razão, cálculo e linguagem". Antes disso, Papert trabalhou cinco anos com Piaget, em seu Centro de Epistemologia Genética, na Suíça.

Prado (1999) afirma que Piaget e seus colaboradores contribuíram efetivamente para a compreensão do desenvolvimento humano e que o construcionismo de Papert inspirou-se, em parte, na psicologia genética de Piaget, "noqual o desenvolvimento cognitivo é um processo de construção e reconstrução das estruturas mentais" (Prado, *op. cit.*, p. 27).

Para Piaget (1972), o sujeito ao agir, desenvolve continuada e progressivamente sua inteligência. Em suas próprias palavras: "[...] o conhecimento não procede, em suas origens, nem de um sujeito consciente de si mesmo, nem de objetos já constituídos (do ponto de vista do sujeito) que a ele se imporiam. O conhecimento resultaria de interações que se produzem a meio caminho entre os dois [...]" (Piaget, 1972, p.14). Almeida (1999), por sua vez, elucida o fato da impossibilidade de se transmitir um conhecimento, reiterando que para Piaget ele é "construído progressivamente por ações e coordenações de ações, que são interiorizadas e se transformam" (p.31).

Esses mecanismos de assimilação e acomodação são introduzidos por Piaget (1972) como pressupostos para a construção do conhecimento. A assimilação está ligada a ação do sujeito sobre um objeto, num processo no qual ele incorpora novas experiências ou informações às já existentes. A acomodação é um movimento em que o sujeito modifica suas estratégias de ação, suas idéias e seus conceitos, em função de novas informações/experiências, gerando, portanto, novas estruturas cognitivas. O movimento equilibrante entre a assimilação e acomodação é de natureza constante e é caracterizado como adaptação, constituindo-se como um dinamismo fundamental ao desenvolvimento cognitivo.

> O sujeito inserido num certo contexto histórico, político, social, realiza reflexões sobre a sua ação, ou seja, o sujeito apropria-se de sua ação, analisa-a, retira elementos de seu interesse e a reconstrói em outro patamar. A ação material do sujeito e suas possíveis evocações propiciam abstrações empíricas, enquanto que as abstrações reflexivas resultam das coordenações das ações do sujeito. (ALMEIDA, 1999, p.32)

Percebe-se, então, um indicativo da sintonia da teoria da aprendizagem de Jean Piaget e o pensamento de Papert, que tenta inseri-la em um ambiente informatizado. Como visto anteriormente, durante a atividade de programação de computadores – base de ação construcionista – acontece o processo de reflexão e o de depuração. Nesse sentido, a reflexão propiciaria a assimilação de conceitos ligados à resolução de problemas usando uma linguagem de programação. Já a depuração, proporcionaria a acomodação do conhecimento, por meio da revisão de estratégias de solução de problemas, as quais seriam reelaboradas em níveis de compreensão superiores.

Reitera-se, que num processo onde o sujeito é ativo na construção de seu conhecimento – como é o caso da programação de computadores –, deseja-se que este, ao agir, que o faça de forma consciente. Isso significa que esse sujeito não deve ser apenas um executor de tarefas, ao contrário, deve compreender aquilo a que se propõe a realizar. Mais uma vez, tem-se em Piaget o respaldo para essa afirmação:

> [...] fazer é compreender em ação uma dada situação em grau suficiente para atingir os fins propostos, e compreender é conseguir dominar, em pensamento, as mesmas situações até poder resolver os problemas por elas levantadas, em relação ao porquê e ao como das ligações constatadas e, por outro lado, utilizadas na ação (PIAGET, 1978, p. 176).

Dessa maneira, alcançar o sucesso em uma atividade realizada não implica que o aluno tenha conseguido compreender aquilo que realizou. Essa constatação tem implicações diretas na aprendizagem, a qual deve visar, preponderantemente, a compreensão. Valente (1999a) afirma que Piaget constatou que a compreensão está intrinsecamente ligada à qualidade da interação entre o sujeito e o objeto. Assim, em nosso caso, se um aluno possui a condição de desenvolver um programa de computador, refletir sobre os resultados e encontrar propostas de melhorá-lo, ele tem a chance de alcançar a compreensão. “Não será o fazer, o chegar a uma resposta, mas a interação

com o que está sendo feito, de modo a permitir as transformações dos esquemas mentais" (Valente, 1999a, p.39).

Almeida (1999) ao refletir sobre as idéias de Castorina (1996) a respeito do "fazer e do compreender" segundo Piaget, destaca que esses estão ligados a problemas hauridos no meio social e enfatiza tal aspecto. Assim, esse autor propõe que, apesar da presença das condições sociais na teoria piagetiana, esta não as enfatiza, e propõem que "a internalização cultural estudada por Vygotsky, bem como seu constructo da 'zona de desenvolvimento proximal (ZDP)', podem ser articulados com estudos piagetianos, integrando aspectos cognitivos e sócio-históricos" (Almeida, *op. cit.*, p.34).

Vygotsky (1984) encara o homem como um sujeito integrado com seu meio social. A partir daí, concebe a base do desenvolvimento do indivíduo como resultado de um processo social e histórico, onde a linguagem desempenha um papel fundamental. Essa última, segundo o teórico, funciona como um instrumento de mediação viabilizador do convívio/contato social e, conseqüente, do desenvolvimento do sujeito. Nesse sentido, Porto Alegre (2005) afirma que para Vygotsky os processos psicológicos superiores deveriam ser vistos como produto de uma atividade mediada e indica Oliveira (1997) para uma melhor compreensão dessa afirmativa: "mediação, em termos genéricos, é processo de intervenção de um elemento intermediário numa relação; a relação deixa então de ser direta e passa a ser mediada por esse elemento" (Oliveira, *op.cit.*, p. 26). Dessa forma,

> Para compreender o indivíduo, é necessário compreender as relações sociais que se estabelecem no ambiente em que ele vive. Isto significa compreender as relações entre atividade prática e trabalho, no sentido de que a atividade prática é transformadora e institucionalizada, envolve dialética ente o trabalho manual e os processos comunicativos. Atividade prática não se restringe à ação sobre os objetos, mas, sobretudo, ao posicionamento do homem em relação ao mundo historicamente organizado. (ALMEIDA, 1999, p.35)

Assim, as atividades práticas constituem-se como oportunidades para a interação entre os sujeitos em seus meios sociais. Percebe-se, portanto, que além de ativo o sujeito passa a ser interativo no que tange seu desenvolvimento, que é consolidado por intermédio de relações com os outros, viabilizadas – essencialmente – pela linguagem.

Outra importante contribuição de Vygotsky está ligada à aprendizagem. O teórico relacionou a aprendizagem ao desenvolvimento, instituindo o conceito de “Zona de Desenvolvimento Proximal” (ZDP), a qual seria a "distância entre o Nível de Desenvolvimento Real (NDR), que se costuma determinar através da solução independente de problemas, e o Nível de Desenvolvimento Potencial (NDP), determinado através da solução de problemas sob a orientação de um adulto ou em colaboração com companheiros mais capazes" (Vygotsky, 1984, p.97). Em outras palavras, seria a diferença entre o desempenho independente e o desempenho assistido.

A identificação da ZDP de um aluno representa, para um professor, a oportunidade do acesso à maturação da aprendizagem de seu aluno. Assim, a ZDP caracteriza-se como a propícia para a mediação, que, ao ocorrer fora de seus limites, incorreria em duas situações de ineficácia: ou o educando já dominaria o que lhe é proposto, ou ele não seria capaz de se apropriar daquilo que lhe é apresentado.

Almeida (1999) indica que Papert retoma de Vygotsky a importância dos signos, essencialmente, a linguagem. Sem ela, a interações estabelecidas entre aluno-aluno, aluno-professor, aluno-computador não se processariam, o que inviabilizaria a construção do conhecimento. Mais uma vez, a figura do professor tem a oportunidade de contribuir para a promoção de uma aprendizagem significativa para seus alunos. Em suas intervenções dentro do processo DERD, o profissional da educação deve se esforçar para atuar dentro da ZDP dos alunos e, fundamentalmente, não se furtar aos debates, à pesquisa em conjunto e ao fomento do trabalho cooperativo.

Ao enfocar-se o meio social no qual um sujeito está inserido, admite-se que ele possa oferecer subsídios para o desenvolvimento intelectual e também representar uma fonte de problemas contextuais que demandam soluções. Valente (1993b) aponta, neste fato, uma ligação como o pensamento do educador brasileiro Paulo Freire: “o aluno pode aprender com a comunidade bem como auxiliar a comunidade a identificar problemas, resolvê-los e apresentar a solução” (Valente, 1993b, p. 45).

Paulo Freire representou o antagonismo da visão tradicional da educação baseada unicamente na transmissão de conhecimentos e memorização de conteúdos. Em sua crítica a essa prática, defendeu uma proposta na qual a visão de “ensinar” não seria resumida a “transferir conhecimento, mas criar as possibilidades para a sua própria

produção ou a sua construção" (Freire, 1996, p.27). Tem-se, portanto, uma proposta educadora que incorpora em suas diretrizes, a leitura de mundo do educando, sua visão crítica da realidade, a formação de um amálgama entre teoria e prática e que busca conferir a seus sujeitos elementos para o exercício de sua criticidade e autonomia. De outra forma, o processo educativo não estaria cumprindo com suas prerrogativas e estaria formando sujeitos vazios, conforme caricaturado na figura 2.13.

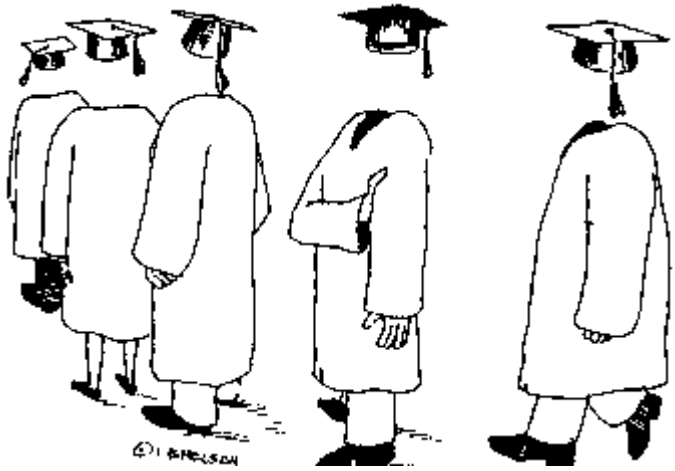

Figura 2.13 – *Empty graduates,* por Ibnelson.
Fonte: http://cavaleiroerrante.blogdrive.com acesso em 19/02/2009

Papert (1993/2008), nos elementos pré-textuais de seu livro "A máquina das crianças: repensando a escola na era da informática", revela que seu pensamento naquela obra foi enriquecido em conversações com pensadores, entre eles, o educador brasileiro Paulo Freire. Papert retomou de Freire a crítica à educação bancária (Almeida, 1999). Em um encontro entre os dois, realizado na Pontifícia Universidade Católica – PUC/SP, em novembro de 1995[6], Papert iniciou suas palavras contando uma anedota que explorava exatamente o vazio proporcionado pelo sistema educativo baseado, exclusivamente, na instrução. Para o autor sul-africano a idéia de o aluno tornar-se sujeito de seu próprio processo de aprendizagem, deveria ser concretizada, representando uma forma alternativa ao modelo vigente. Nesse sentido, o uso do

[6] Esse debate pode ser assistido na fita de vídeo cujo nome é "O Futuro da Escola" da TV PUC, São Paulo, 1995.

computador como ferramenta coloca o educando no controle do processo educacional, possibilitando uma readequação de sua posição de consumidor de informações.

É importante, ainda, se destacar o aspecto afetivo dentro do processo de interação, no qual uma atividade proposta deve ter relevância para o educando, visando seu engajamento. Papert (1993/2008) demonstra concordância em relação a esse pensamento, ao descrever uma experiência com alunos concluintes do ensino fundamental relacionada à coleta de dados sobre a chuva ácida e a transmissão desses usando redes eletrônicas visando uma análise posterior:

> O projeto sugere um cenário de milhões de crianças em todo o mundo engajadas em um trabalho que oferece uma contribuição real para o estudo científico de um problema social urgente. Em princípio, um milhão de crianças poderia coletar mais dados sobre o ambiente do que qualquer número socialmente custeável de cientistas profissionais.
>
> Isso é muitíssimo melhor do que folhas de exercício e experimentos ritualísticos da escola, pois pelo menos os aprendizes sentem que estão engajados em uma atividade significativa e socialmente importante, sobre a qual eles concretamente se sentem responsáveis. (PAPERT, *op. cit.,* p.38)

Papert (1986) expressa sua percepção pessoal da importância da criação de ambientes de aprendizagem que proporcionem oportunidades de ampliar a qualidade das interações referentes ao que está sendo realizado. É o princípio do *hands-on*/*head-in*. Como sujeitos ativos da dinâmica educacional, os aprendizes precisam "colocar a mão na massa" (*hands-on*) no desenvolvimento de suas atividades, em um movimento contrário ao de serem espectadores dos discursos de seus professores. Assim, a aprendizagem se concretiza no momento em que seus sujeitos tornam-se construtores conscientes e ativos de um "produto público", que tenha relação com o contexto social onde eles estejam inseridos e que, essencialmente, possuam interesse pessoal em concretizar (*head-in*).

O construcionismo de Papert é fruto de um desejo pessoal em promover um processo de aprendizagem rico de significados para os sujeitos que dele participam. Iniciativas, necessidades, interesses, pesquisa, reflexão, desenvolvimento crítico, incentivo à criatividade e colaboração são alguns dos elementos presentes na abordagem de Papert que, unidas ao uso do computador, configuram uma alternativa ao tradicional processo de transmissão de conhecimento.

CAPÍTULO III

LOGO: CONCRETIZAÇÃO DO CONSTRUCIONISMO

O LOGO foi incentivado, desde o início, por uma perspectiva tipo "Robin Hood" de roubar a programação dos tecnologicamente privilegiados.
Seymour Papert

Uma das características da proposta construcionista está ligada ao que ficou caracterizado no capítulo anterior por *hands-on/head-in*. Para seu idealizador – Seymour Papert – essa perspectiva constitui-se ao longo de sua trajetória de vida e foi baseada em inúmeros projetos, dentre os quais, se destaca a ferramenta educacional LOGO.

Este capítulo apresenta a ferramenta LOGO, explorando sua origem e evolução, seus aspectos computacionais e pedagógicos.

3.1 – Origem e evolução

Em Papert (1993/2008), ao longo do capítulo intitulado "Computadoristas", o autor relata a história de criação Linguagem de Programação LOGO. Nele, percebe-se que na origem dos computadores – em meio à década de 40 e ao esforço de guerra – os primeiros protótipos foram destinados a cálculos complexos e decifração de códigos. Seus usuários "pioneiros eram matemáticos e construíram máquinas à própria imagem". O autor relata ainda que aquela época/situação inaugurava uma "cultura de computadores sem qualquer espaço para o pluralismo" e denomina-a de "cultura de rígidos (*hards*)" (Papert, 1993/2008, p.149).

> Naquela época era necessário que se espremesse a última gota de força da máquina para se fazer até mesmo trabalhos muito simples, e isso, com freqüência, significava efetuar, mentalmente, malabarismos com matemática computacional. Lembro-me de minhas primeiras experiências de programação assemelhando-se mais à resolução de problemas em teoria dos números do que à atividade auto-expressiva [...]. O que quero ressaltar não é simplesmente que aquilo era uma cultura matemática (o que de fato era), mas um tipo particular de cultura matemática no qual o cálculo acurado desempenhava o papel dominante, e o técnico e o analítico tinham mais peso do que o intuitivo e o experimental.
>
> Assim, muitos fatores contribuíram para moldar a cultura inicial do computador na rígida e analítica forma que, para a maioria das pessoas, até mesmo hoje, permanece sinônimo da palavra computador. [...]
>
> [...] Quando programei o ACE [*Automatic Computer Engine*, lançado em 1946], tive que expressar instruções como seqüências de 0 e 1, literalmente codificados por meio da perfuração de orifícios, um a um, em um cartão IBM. [...]
>
> [...] Expressar instruções com números binários é algo muito opaco e entediante para que até mesmo um matemático considere confortável. (PAPERT, 1993/2008, p. 150)

A aplicação do computador em domínios mais amplos sucedeu a Segunda Grande Guerra. A pesquisa industrial e a universitária fomentaram essa expansão, que foi estendida, posteriormente, para outros campos do conhecimento. Ainda na década de 60, Papert acusa o surgimento da idéia de se utilizar computadores na Educação. Ele fazia parte de um grupo de "atores desconhecidos" que buscava implantar uma "cultura de computadores especificamente educacional" (Papert, 1993/2008, p. 152). No trecho citado, Papert faz referência a um grupo de professores: Patrick Suppes (Filosofia/Psicologia) mentor intelectual do CAI; Jonh Kemeny (Física), idealizador da linguagem de programação BASIC; Donald Bitzer (Engenharia) proponente do PLATO - *Programed Logic for Automatic Teaching Operation* – um sistema de autoria no qual um professor poderia criar suas próprias aplicações CAI.

Papert (1993/2008), em sua concepção do uso do computador na educação, mostrava-se, ainda na década de 60, desejoso de implantar uma cultura mais suave (*softer*). Baseado nesse desejo e sendo contrário ao *software* do tipo CAI, o autor buscava um caminho para concretizar seus ideais.

> Isso ocorreu em 1965, em uma visita à Ilha de Chipre. Eu ainda estava meio tonto, devido ao choque cultural por ter-me transferido (em 1963) da Universidade de Genebra [quando trabalhou com Piaget], onde não havia computadores, para o MIT, onde de repente tive acesso às melhores máquinas do mundo. Lá naquela remota ilha do Mediterrâneo, senti a ausência de um estilo de vida no qual os computadores fossem uma presença constante. Isso, por sua vez, me fez refletir sobre o quanto eu havia aprendido desde que viera para o MIT; como havia usado o computador para avançar sobre um problema teórico que me incomodava durante algum tempo, como os conceitos relacionados aos computadores estavam mudando o meu pensamento em muitas áreas diferentes. Então, em um flash, surgiu a idéia "óbvia": o que os computadores proporcionaram a mim era exatamente o que deveriam proporcionar às crianças! Eles deveriam servir às crianças como instrumentos para trabalhar e pensar, como meios para realizar projetos, como fonte de conceitos para novas idéias. A última coisa no mundo que eu desejava ou precisava era de um programa de exercício e prática dizendo-me para fazer uma soma ou escrever uma certa palavra! [...] Tornei-me obcecado pela pergunta: Poderia o acesso a computadores permitir às crianças algo semelhante ao impulso intelectual que experimentei com o acesso aos computadores do MIT? (PAPERT, *op. cit.*, p. 158)

A resposta para o auto questionamento de Papert constitui-se como a base de sua proposta construcionista, sua "marca registrada" na forma de usar os computadores na Educação, ou seja: o *software* LOGO. Paradoxalmente, em uma cultura de *hards*, Papert pretendia proporcionar às crianças uma maneira de explorar o universo da computação de uma forma que lhes fosse inteligível, acessível e benéfica. Mas, como fazê-lo?

A idéia de programação, dentro da cultura vigente, não se aplicaria em hipótese alguma a um processo educacional com crianças. As linguagens de programação da época exigiam um grau de sofisticação matemática inapropriado às crianças e, possivelmente, teriam sido planejadas para o uso adulto. Dessa maneira, o desafio era construir uma forma alternativa ao que se praticava com os computadores na Educação (CAI), romper com uma cultura de *hards* e possibilitar o acesso aos recursos computacionais disponíveis à aprendizagem mesmo a pequenos aprendizes. Papert conviveu com essas indagações com a clara certeza de que, de alguma maneira, uma cultura *softer* precisava ser "inaugurada".

O caminho seguido por ele e seu grupo de apoiadores no MIT foi consolidado ainda nos meados da década de 60, sob o formato de uma linguagem de programação que tinha um vocabulário simples e fácil de ser aprendido. Em suas

primeiras versões, o LOGO não contava com uma parte gráfica e era utilizado por crianças de idade compatível ao ensino fundamental no Brasil. A base de sua aplicação relacionava-se à escrita de poesias, elaboração de jogos de estratégia, entre outros. Esse fato demonstrava que o LOGO era de fácil aprendizagem.

O objetivo de Papert era estender o uso daquela ferramenta ao nível de crianças da pré-escola (Papert, 1980/1985). Para isso, novas adequações deveriam ser implementadas a fim de prover o acesso ao *software* para esse novo público.

> A idéia demorou um pouco para surgir e um tempo ainda mais longo para que eu percebesse o seu significado. No começo eu estava bloqueado ao procurar obstinadamente por algo demasiadamente novo, de uma maneira que ocorre com freqüência. Depois disso, percebi que durante todo o tempo eu tinha a solução para o problema, mas não conseguia vê-la porque estava forçando o modo de ver e tensionando a mente ao procurar lá fora, até onde a vista pudesse alcançar. Encontrei a solução quando parei de levar-me tão à sério e de procurar tão intensamente algo novo. A nova idéia surgiu quando olhei de uma forma mais relaxada para algo que estava à mão.
>
> Eu estava rabiscando no computador, como tantas vezes faço, escrevendo pequenos programas sem nenhuma importância particular ou dificuldade em si mesmos. Poderíamos chamar isso de simples brincadeira. [...] O que ocorreu dessa vez resultou de pensar que escrever programas pode ser semelhante, de muitas formas, a desenhar. (PAPERT, 1993/2008, p.163)

Aquele *insight* rendeu novas reflexões, entre elas que desenhar e caminhar eram coisas típicas do universo infantil. Daí em diante, o esforço era buscar estabelecer um elo entre uma forma computacional e algo físico que pudesse desenhar/caminhar. “A resposta foi um robô amarelo com uma forma bastante semelhante à do R2D2[7] e, como ele, montado sobre rodas” (Papert, 2008, p.164), o que pode ser observado na figura 3.1.

[7] R2D2 é um dos robôs do filme *Star Wars* (Guerra nas Estrelas).

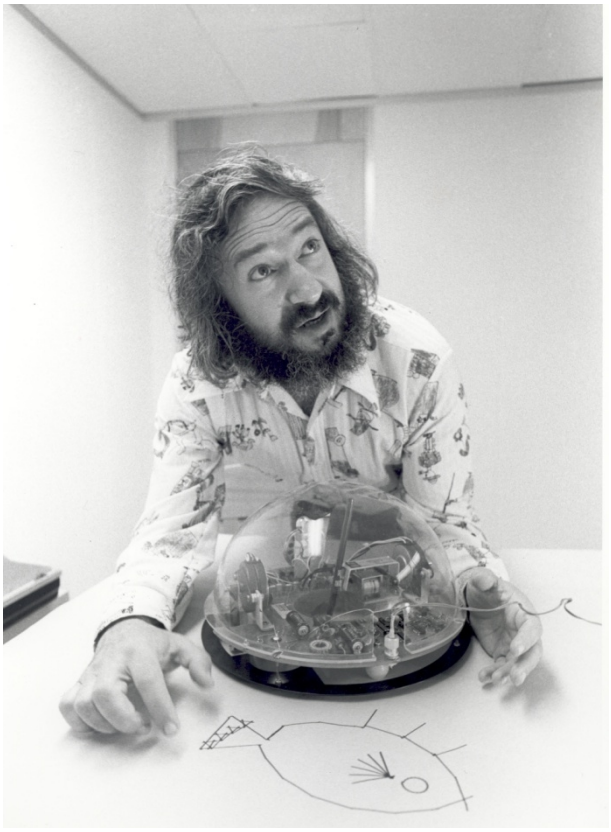

Figura 3.1 – Papert e o robô Tartaruga.

Papert concretizou suas idéias ao construir um *hardware* – a tartaruga robô – que recebia e executava ordens de um usuário do *software* LOGO. "Podia-se mandá-la andar dando algumas instruções em LOGO gramaticalmente adequadas" (Papert, 1993/2008, p.164) e ao fazê-lo, essa deixava um rastro que formava uma figura. A programação com a linguagem LOGO assumiu uma nova dimensão, assemelhava-se à construção de figuras geométricas seguindo os comandos nativos da ferramenta.

> Eu propus a Tartaruga como uma área de programação que poderia ser interessante a todas às idades. Toda essa expectativa tem sido confirmada pela experiência, e a Tartaruga, como instrumento de aprendizagem, tem sido amplamente aceita e adotada. Um trabalho pioneiro do uso de Tartarugas para ensinar crianças bem pequenas foi realizado por Radia Perlman que demonstrou, enquanto era estudante do MIT, que crianças de quatro anos podiam aprender a comandar Tartarugas mecânicas. [...] Na outra extremidade do leque de idades, é encorajador ver que a programação com a Tartaruga está sendo usada a nível universitário para ensinar PASCAL. (PAPERT,1980/1985, p. 26)

Finalmente, o LOGO estava adequado ao nível infantil e com ele surgia a possibilidade de estender, com pluralidade, o domínio de uso do computador. Esse fato

consolidou a proposta de Seymour Papert como uma alternativa ao modelo de uso dos computadores na Educação baseada na transmissão de conteúdos que, até então, era vigente.

3.2 – LOGO na perspectiva computacional

Do ponto de vista computacional, a Linguagem de Programação LOGO foi concebida para ser inteligível e, dessa forma, de fácil assimilação para iniciantes. Como em qualquer LPC, LOGO também possui vocabulário próprio, o qual é utilizado na construção dos programas.

O ambiente de programação LOGO tipicamente disponibiliza a Tartaruga[8], uma criatura robótica, que é direcionada por meio de comandos da Linguagem de Programação. Desde sua introdução, a linguagem LOGO foi implementada em diferentes *softwares*, cada qual com sua particularidade. Atualmente, esses ambientes possuem uma "Janela de Comandos", que é uma área de interação para o usuário usada para a escrita de comandos usando a Linguagem, bem como a "Janela Gráfica" – o *habitat* virtual da Tartaruga – nela os comandos propostos podem ser executados e seus efeitos observados.

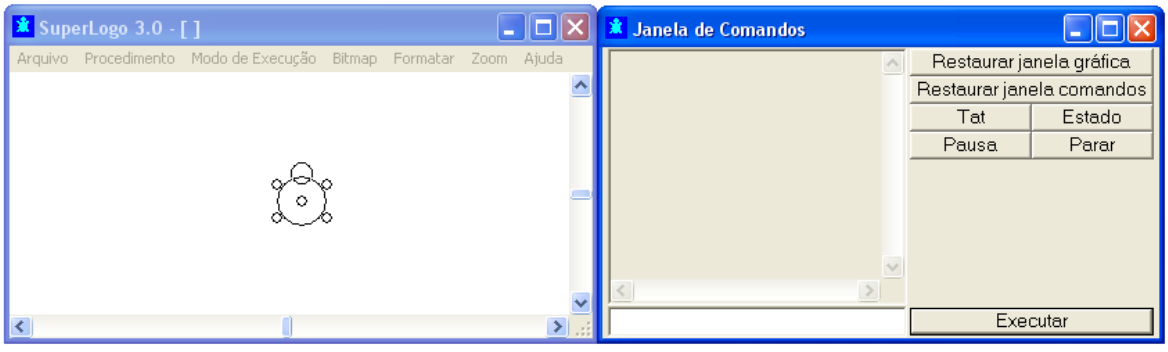

Figura 3.2 - O ambiente do SuperLogo: a esquerda a "janela gráfica" e a direita a "janela de comandos".

A figura 3.2 mostra o *software* SuperLogo, versão que foi desenvolvida por George Mills e Brian Harvey, da Universidade de *Berkeley*, que foi traduzida e adaptada pela equipe do Núcleo de Informática Aplicada à Educação da Universidade Estadual de

[8] A Tartaruga pode ser um objeto virtual que se desloca nos monitores de vídeo dos computadores, ou, como na sua origem, um objeto físico como os robôs que se deslocam no chão.

Campinas (UNICAMP), podendo ser livremente copiada e distribuída, pois é regida pela GNU.

Valente (1993, p. 19) afirma que “a exploração de atividades espaciais tem sido a porta de entrada do LOGO”. Essas atividades envolvem, intuitivamente, conceitos espaciais que são próprios à infância. Assim, o vocabulário básico da linguagem LOGO, diz respeito ao deslocamento (caminhar) da Tartaruga para diferentes pontos da tela. A tabela 3.1 apresenta esses comandos.

Tabela 3.1 - Comandos básicos do LOGO seguidos de suas descrições e exemplos.

Comando	Descrição	Ex.	Efeito na tela
parafrente <número> ou **pf <número>**	Desloca a Tartaruga para frente um determinado número de passos.	**pf 50**	
paratrás <número> ou **pt <número>**	Desloca a Tartaruga para trás um determinado número de passos.	**pt 50**	
paradireita <numero> ou **pd <número>**	Gira a Tartaruga à direita um determinado ângulo.	**pd 90**	
paraesquerda <número> ou **pe <número>**	Gira a Tartaruga à esquerda um determinado ângulo.	**pe 90**	

Em Prado (1999), verifica-se que esses comandos podem ser inicialmente explorados por um usuário utilizando o “modo direto”. Nessa modalidade, a cada comando digitado pelo usuário, a Tartaruga realiza a ação cabível. As figuras 3.3A até a 3.3H, mostram a seqüência de construção um retângulo utilizando o *software* SuperLogo.

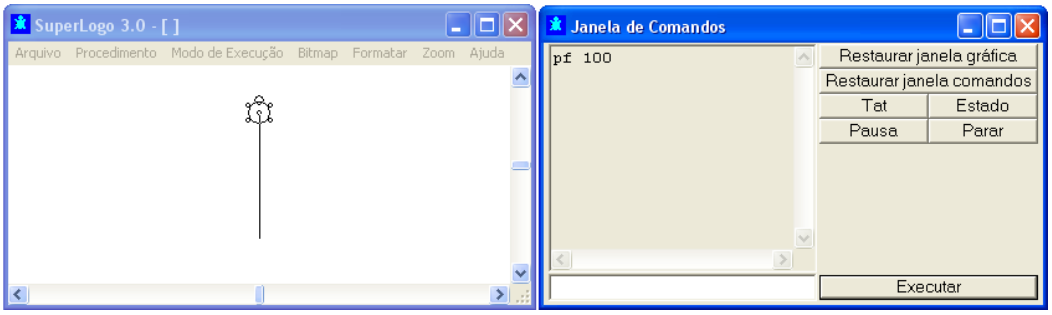

Figura 3.3A – Passo 1 da construção do retângulo no modo direto do SuperLogo.

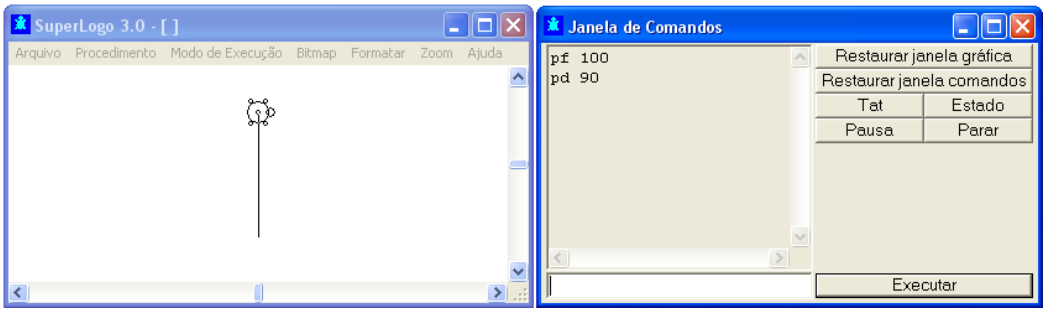

Figura 3.3B – Passo 2 da construção do retângulo no modo direto do SuperLogo.

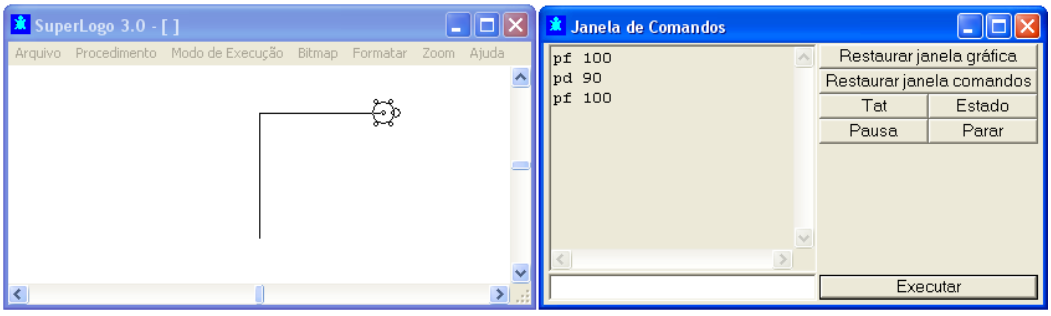

Figura 3.3C – Passo 3 da construção do retângulo no modo direto do SuperLogo.

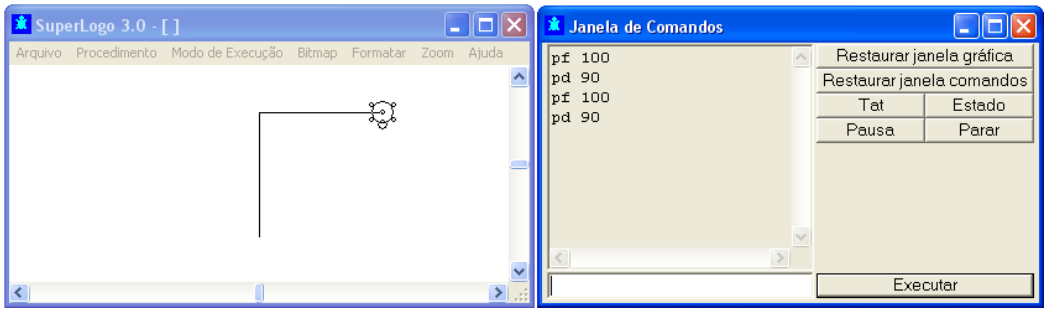

Figura 3.3D – Passo 4 da construção do retângulo no modo direto do SuperLogo.

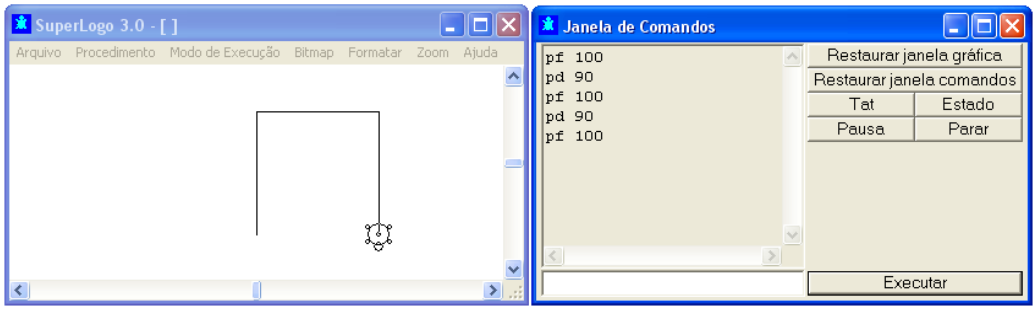

Figura 3.3E – Passo 5 da construção do retângulo no modo direto do SuperLogo.

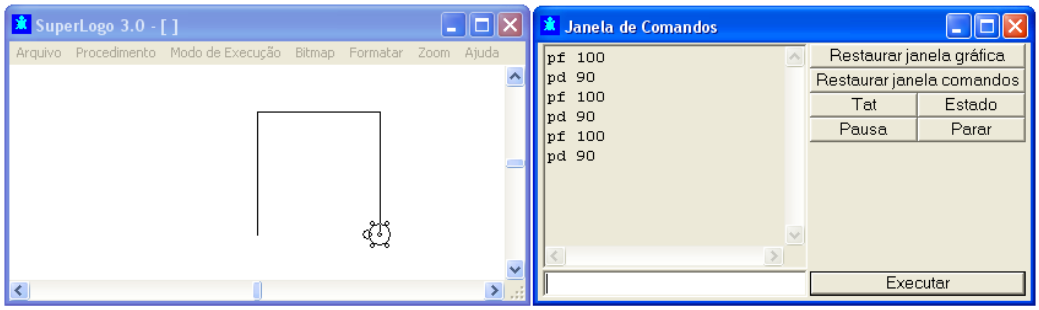

Figura 3.3F – Passo 6 da construção do retângulo no modo direto do SuperLogo.

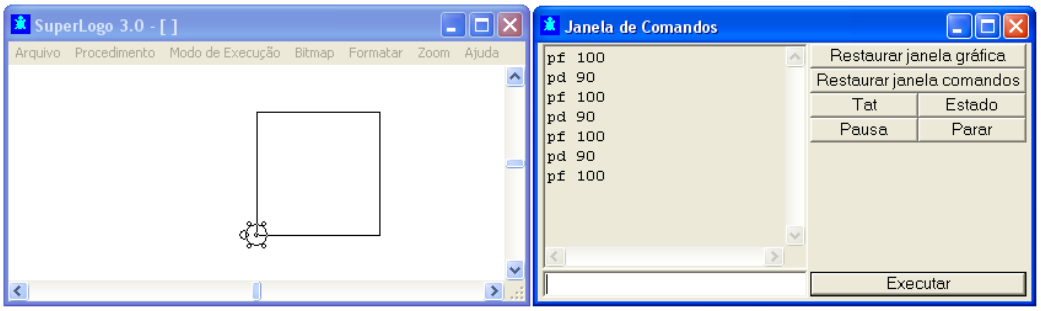

Figura 3.3G – Passo 7 da construção do retângulo no modo direto do SuperLogo.

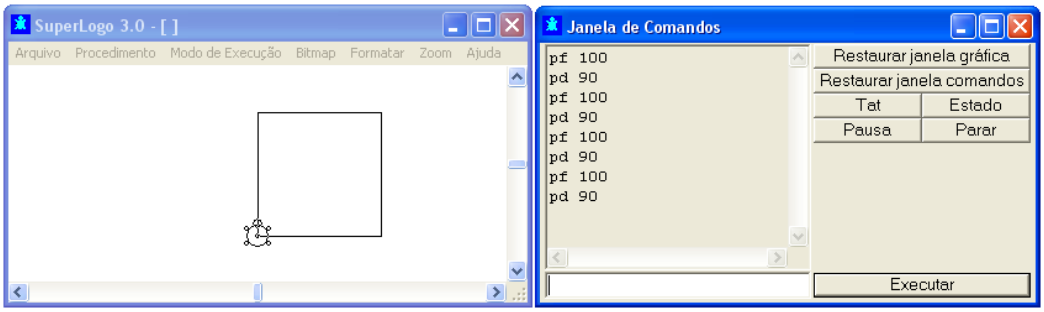

Figura 3.3H – Passo 8 da construção do retângulo no modo direto do SuperLogo.

A construção de figuras no “modo direto” usando os comandos básicos da linguagem LOGO é uma atividade que permite a exploração de propriedades geométricas de uma figura. No “passo a passo” dessa modalidade, o usuário tem a oportunidade de

escrever um comando, a Tartaruga executá-lo, com o rastro deixado por ela o usuário recebe o *feedback* e passa a refletir sobre sua ação. Daí, se tudo correr conforme o pretendido, o aluno continua com sua descrição; na eminência de um equívoco, ele terá a oportunidade de voltar atrás e rever e depurar seus conceitos, apropriando-os. Tem-se, portanto, a aplicação direta do processo de DERD, inerente à atividade de programação de computadores.

A figura 3.3H revela uma forma geométrica inteiramente construída a partir da seqüência de comandos digitada pelo usuário. Particularmente, o que foi comandado pelo usuário foram oito instruções que, em conjunto, formam um programa. Papert (1980/1985, p. 27) ressalta que a "idéia de programação é introduzida através da metáfora de ensinar à Tartaruga uma nova palavra". Isso equivale a dizer que, inicialmente, a Tartaruga não saberia o que era um quadrado, entretanto, um programador aprendiz, poderia se valer dos comandos nativos em LOGO para criar novos termos em seu vocabulário. Isso é conseguido no "modo de edição", que é apropriado para a escrita de procedimentos (novas palavras para o vocabulário da Tartaruga).

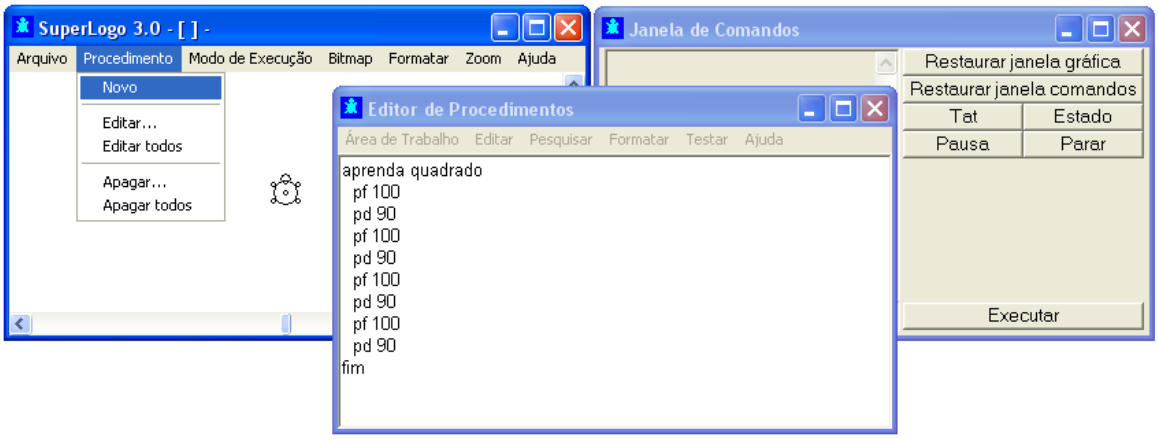

Figura 3.4 –Modo de edição de procedimentos no SuperLogo com o programa quadrado de lado 100.

Na figura 3.4 observa-se a definição do programa "quadrado", que representa dentro do ambiente de programação, um novo termo pelo qual a Tartaruga irá passar a atender. Nota-se que esse "novo proceder" é iniciado pela palavra "APRENDA" seguida do nome do programa (no caso, quadrado). O corpo do programa é formado por comandos básicos do LOGO, logicamente ordenados, objetivando a construção de um quadrado. O Programa é finalizado com a palavra "fim". Após a escrita

do programa quadrado, ao se recorrer ao “modo direto” de execução e comandar o termo quadrado, imediatamente a Tartaruga passa a executar o programa, baseado no que foi determinado pelo programador (figura 3.5).

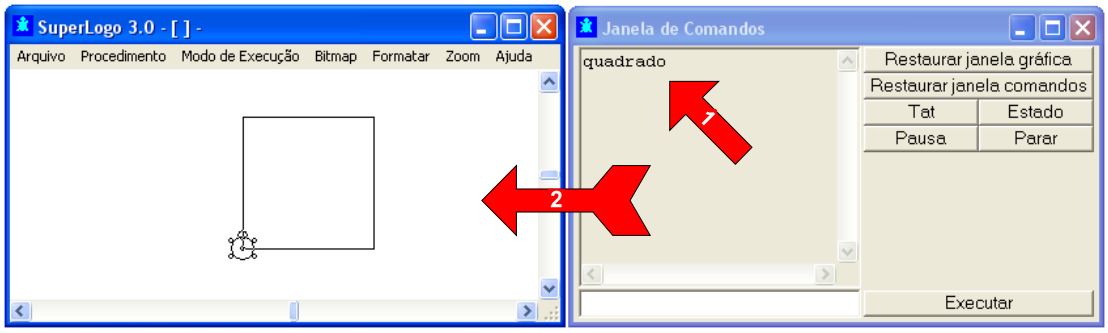

Figura 3.5 – Modo direto reconhecendo o novo termo “quadrado”.

Nota-se que no “modo de edição” o sujeito tem a oportunidade de explorar as mesmas instruções do “modo direto”. Entretanto, ele é levado a antecipar resultados e a construir relações entre cada linha de seu programa, visando à consecução de seu objetivo final. É a forma mais elaborada, em termos cognitivos do ato de se instruir, ou programar um computador. Nas palavras de Papert (1980/1985, p. 22): “Algumas das características fundamentais da família de linguagens LOGO são as definições de procedimentos [...]. Assim, em LOGO é possível definir novos comandos e funções que podem ser usados exatamente como as funções primitivas da linguagem”.

As unidades curriculares que envolvem a programação de computadores compreendem inúmeros conceitos fundamentais, os quais podem ser explorados em diferentes linguagens. Entre esses, destacam-se: variáveis, estruturas de repetição, estruturas condicionais, recursão, passagem de parâmetros, entre outros. Usualmente, no nível universitário, o que insere o uso de uma LPC em um processo de ensino-aprendizagem é exatamente o suporte à implementação desses conceitos.

Como visto, uma LPC serve como meio de expressão de idéias, sendo utilizadas as regras sintáticas e semânticas dos termos da linguagem. A depender da linguagem em uso, a compreensão da forma de descrição de um problema em seu vocabulário pode gerar dificuldade, uma vez que os termos da linguagem são geralmente provenientes da língua inglesa, bem com as mensagens de erro. Além disso, a

complexidade (cultura de *hards*) do modelo de representação de idéias na linguagem pode, também, ser um fator complicador, principalmente para iniciantes. Em contrapartida, se uma linguagem de programação oferecer suporte a esses conceitos e puder ser aplicada, sobre uma "ótica *softer*", em um processo de ensino-aprendizagem, as dificuldades relacionadas a essa dinâmica de aprendizagem podem ser minoradas. Esse é o caso do LOGO.

Demonstra-se a seguir, alguns dos conceitos supracitados, que o LOGO oferece suporte. Para tal, considera-se o exemplo anterior, referente à construção do retângulo. Nele, pode-se notar que o quadrado foi composto por intermédio de oito instruções (ver Figura 3.4). Facilmente, observa-se que o par de instruções "PF 100, pd 90" aparece, seguidamente, quatro vezes. O conceito de estrutura de repetição pode ser utilizado para aprimorar o raciocínio relativo à construção do retângulo. No caso, o comando "REPITA", seguido do número de vezes e do que se deseja repetir, pode ser aplicado.

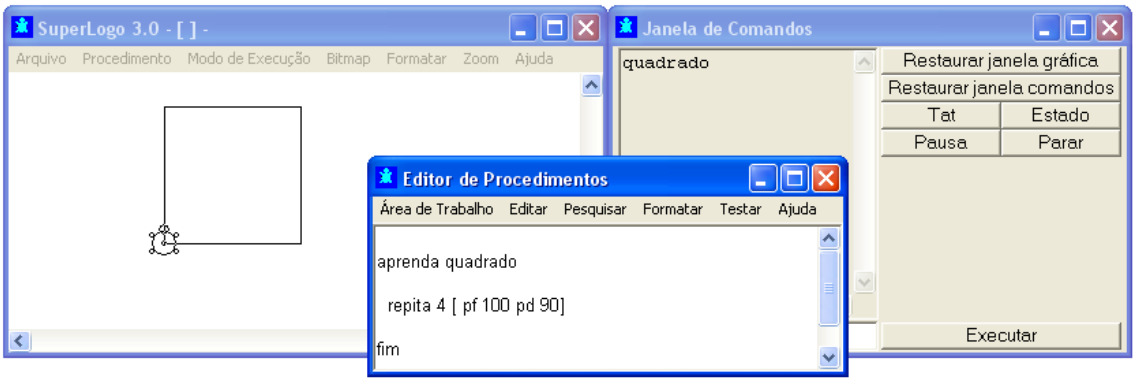

Figura 3.6 – O programa quadrado como conceito de estrutura de repetição.

Outro ponto teórico fundamental em computação diz respeito à passagem de parâmetros e o uso de variáveis. Pode-se entender facilmente esse conceito com a ajuda do LOGO. Nota-se que na Figura 3.6 o quadrado produzido foi exatamente igual (tamanho) ao da Figura 3.5, embora o programa tenha sido expresso em uma forma alternativa. Para o caso da produção de quadrados de tamanhos personalizados, esse valor poderia ser explicitado pelo usuário, junto do termo quadrado. Por exemplo: "quadrado 50" produziria um quadrado de lado 50, ou "quadrado 75", o faria com um

lado de 75. Nesse sentido, estaria-se permitindo à Tartaruga ficar “livre” para desenhar a figura em um tamanho parametrizado.

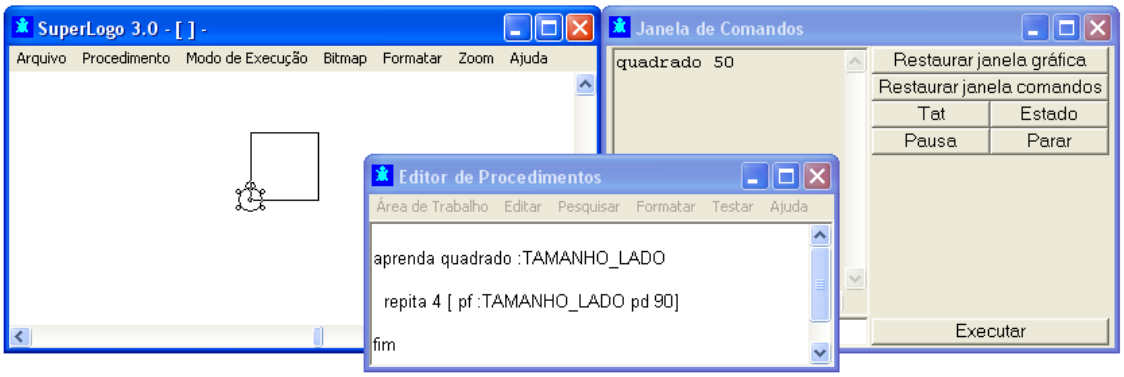

Figura 3.7– O programa quadrado usando passagem de parâmetro e repetição.

Nota-se que na figura 3.7 que foi introduzido o parâmetro “TAMANHO_LADO” junto ao nome do procedimento. Esse parâmetro foi usado como sendo uma variável junto ao comando “pf” (para frente), o que acabou por possibilitar a confecção do quadrado no tamanho especificado pelo usuário do comando.

Após a definição de um procedimento, esse pode ser utilizado tanto no “modo direto” – como no exemplo da figura 3.7 – bem como ser um subprocedimento em outro procedimento. Pode-se exemplificar isso criando um procedimento novo, chamado “giraquadrado”, o qual faz uso do procedimento “quadrado” definido a priori (figura 3.8).

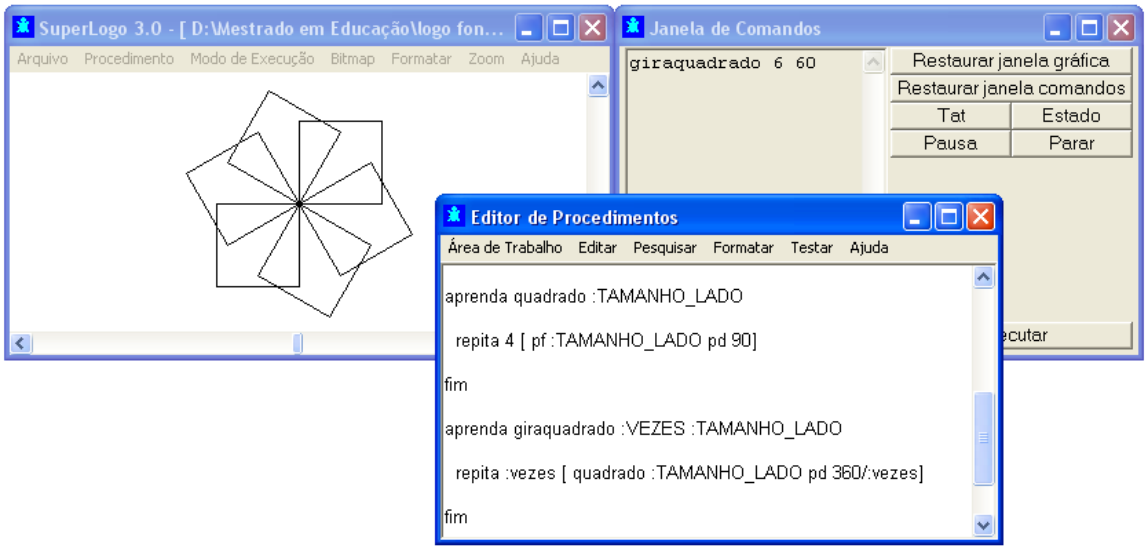

Figura 3.8 – O procedimento quadrado como subprocedimento no procedimento giraquadrado.

Essa característica modular do LOGO é muito significativa em termos computacionais, pois permite o reaproveitamento dos procedimentos já definidos na elaboração de novos procedimentos.

Para finalizar esta breve demonstração do valor computacional da linguagem LOGO será apresentado, a seguir, um pequeno programa utilizando o conceito de recursão. Para isso, propõe-se a criação de uma espiral. Foi visto que um procedimento existente pode ser utilizado como parte de outro procedimento. A recursão está ligada ao uso de um procedimento dentro de si próprio.

Figura 3.9 – Construção de uma espiral com recursão.

Para a compreensão do procedimento “espiral” exibido na figura 3.9, primeiramente destaca-se o uso de duas variáveis: “LADO” e “ÂNGULO”. Essas irão, em conjunto, determinar a forma da espiral (abertura). Na figura indicada anteriormente, nota-se que o usuário escolheu os valores iniciais “1” e “20” para as variáveis “LADO” e “ÂNGULO”. À medida da execução do programa, a variável “LADO” vai sendo incrementada recursivamente, ou seja, seu valor vai crescendo, o que permite a expansão da espiral. A estrutura condicional “**se (:LADO > 50) [pare]**” é muito importante para o entendimento do programa em questão. Ela pode deve ser entendida como “**Se o valor de LADO superar 50, então pare de processar a espiral**”. A Tabela 3.2 mostra que esse comando servirá como um critério de parada para a Tartaruga. Sem ele, o processamento

do programa implicaria em uma recursão infinita, que é conhecida computacionalmente como *loop* infinito. A recursão se aplica a esse programa, pois dentro do procedimento espiral, existe uma chamada para o próprio procedimento espiral. É nessa linha, que a variável “LADO” é incrementada de 0,5 em 0,5 unidades. A tabela 3.2 mostra as iterações produzidas para o processamento da espiral.

Tabela 3.2 – Representação do processamento do programa Espiral.

Iteração	Lado	Ângulo	Lado > 50
1	1	20	Falso, então a Tartaruga Desenha
2	1,5	20	Falso, então a Tartaruga Desenha
3	2	20	Falso, então a Tartaruga Desenha
4	2,5	20	Falso, então a Tartaruga Desenha
5	3	20	Falso, então a Tartaruga Desenha
6	3,5	20	Falso, então a Tartaruga Desenha
7	4	20	Falso, então a Tartaruga Desenha
8	4,5	20	Falso, então a Tartaruga Desenha
9	5	20	Falso, então a Tartaruga Desenha
10	5,5	20	Falso, então a Tartaruga Desenha
11	6	20	Falso, então a Tartaruga Desenha
		[...]	
90	45,5	20	Falso, então a Tartaruga Desenha
91	46	20	Falso, então a Tartaruga Desenha
92	46,5	20	Falso, então a Tartaruga Desenha
93	47	20	Falso, então a Tartaruga Desenha
94	47,5	20	Falso, então a Tartaruga Desenha
95	48	20	Falso, então a Tartaruga Desenha
96	48,5	20	Falso, então a Tartaruga Desenha
97	49	20	Falso, então a Tartaruga Desenha
98	49,5	20	Falso, então a Tartaruga Desenha
99	50	20	Falso, então a Tartaruga Desenha
100	50,5	20	Verdadeiro, então a Tartaruga PARA

Com o uso da recursão, nota-se que o programa foi submetido a um ciclo de 100 iterações, que acabaram por produzir a espiral apresentada anteriormente na

figura 2.23. Fundamental atentar para o fato de que, a cada recursão, a variável LADO foi incrementada, produzindo o efeito de expansão da espiral. Em outras palavras, no início, o valor da variável “LADO” era de “1”, no fim de processamento, seu valor atingiu “50,5”, perfazendo 100 iterações com 99 deslocamentos da Tartaruga.

Todos os conceitos aqui explorados o são, igualmente, nas linguagens PASCAL, JAVA, C, C++, PROLOG, entre outras. Como já afirmado anteriormente, o que muda é o vocabulário, o ambiente de programação e a forma subseqüente de representação das idéias em cada linguagem. Dessa maneira, do ponto de vista educacional, o LOGO não fica devendo em nada para as outras linguagens, uma vez que possui recursos avançados para a exploração dos conceitos relacionados à programação de computadores. Pelo contrário, o seu aspecto gráfico aliado a sua simplicidade sintática/semântica, possibilita a rápida visualização de um determinado conceito o que é definitivo para a compreensão.

> Assim LOGO não é um “brinquedo”, uma linguagem apenas para crianças. Os exemplos mais simples [...] mostram algumas maneiras em que LOGO é especial por ter sido planejada para fornecer muito facilmente e bastante cedo acesso à programação de computadores para principiantes [...]. Logo não é um brinquedo, mas uma poderosa linguagem [...]. (PAPERT, 1980/1985, p.22)

Das palavras de Papert, colhe-se a impressão que o LOGO oferece a possibilidade da exploração computacional, em qualquer que seja o nível educacional praticado. E mais, seu uso pode ser aplicado como uma referência inicial aos conceitos de programação para os outros ambientes de programação. Entretanto, LOGO não encerra apenas um meio computacional. Ele agrega um aspecto pedagógico o qual, doravante, será explorado.

3.3 – LOGO e suas possibilidades pedagógicas

Ao se descrever as bases teóricas de Papert, foi salientada a inspiração de Jean Piaget, Vygotsky e Paulo Freire em seu trabalho. Agora, ao serem destacadas as possibilidades pedagógicas oferecidas pelo LOGO, retomam-se alguns pontos daquela discussão.

A fim de se explorar o universo pedagógico ligado ao LOGO, inicialmente, destaca-se de Valente (1993 b, p.44) o seguinte questionamento: “Por quê é necessário um termo para definir o tipo de aprendizagem que acontece no ambiente LOGO ou, mais precisamente, com o LOGO gráfico?”. O autor faz referência ao termo introduzido por Papert (1980/1985) – construcionismo – comparando-o ao construtivismo proposto por Piaget, uma vez que em ambos pensadores, o sujeito é tido como construtor ativo de conhecimento.

> Entendemos "construcionismo", como, incluindo, mas indo além, ao que Piaget chamaria de "construtivismo". A palavra com o 'v' expressa a teoria de que o conhecimento é construído pelo aluno, sem auxílio do professor. A palavra com o 'n' exprime a idéia mais adequada, que isso acontece, principalmente, quando o aluno está envolvida na construção de algo externo/concreto [...] um castelo de areia, uma máquina, um programa de computador, um livro. (PAPERT, 1990, p.3 - livre tradução)

Nota-se que Papert se embasou na teoria construtivista de Piaget para concretizar o construcionismo e, ao fazê-lo, incluiu em abordagem educacional a proposta de uma aprendizagem situada e estimuladora, que estabelece um constante diálogo com o meio social onde acontece (Freire) e que é baseada em cooperação e interação com outros sujeitos (Vygotsky).

O inicialmente proposto por Piaget em termos de interação sujeito e objeto, ganha uma nova dimensão em Papert. Com a introdução do computador, essa interação passa a se perpetuar, geralmente, com o uso de uma linguagem de programação – no caso o LOGO. A construção do conhecimento continua a estar relacionada à interação sujeito e objeto, entretanto, um novo elemento é introduzido no processo: uma linguagem de programação, veículo de expressão de idéias do qual o aluno pode se apropriar para gerar conhecimento.

No construcionismo, a idéia de interação é expandida ao se introduzir no processo a mediação. Valente (1993b) afirma que para Piaget a interação sujeito objeto era observada por um experimentador, que se valia de um método para entender as construções mentais de um sujeito. Entretanto, o “experimentador não é professor e, portanto, ele não tem por objetivo prover ou facilitar a aprendizagem” (p. 44). Inicialmente, Papert não atribuiu a adequada relevância ao papel do professor em teoria; por outro lado, em seu livro “A máquina das crianças”, o educador dedica um capítulo

exclusivamente ao professor, o que demonstra seu reconhecimento de que esse sujeito se constitui em elemento indispensável no ambiente de aprendizagem. Efetivamente, para a implementação da proposta construtivista com a atividade de programação – com o LOGO –, o profissional da educação tem seu lugar definido: buscar compreender as idéias de seus alunos e, após, intervir adequadamente para cooperar com a aprendizagem de seus alunos. Isso vem ao encontro às contribuições de Vygotsky e sua ZDP, como visto anteriormente.

O LOGO dá, ainda, abertura para que o erro seja encarado, pedagogicamente, como uma importante oportunidade de revisão de conceitos e funciona como um catalisador da aprendizagem. Ao elaborar um programa de computador com o LOGO, seguindo o processo de DERD, o aluno tem a oportunidade de, na eminência de um equívoco, enveredar em um processo de reflexão sobre o seu pensar e depurá-lo. É o que Papert (1980/1985) chamou de estratégia de depuração de erros (*debugging)*. "Os erros são benéficos porque nos levam a estudar o que aconteceu, a entender o que aconteceu de errado, e, através do entendimento, corrigi-los" (Papert, *op. cit.*, p.144).

Finalmente, o LOGO prioriza a aprendizagem numa visão antagônica do que acontece com um *software* do tipo CAI, o qual supervaloriza o ensino. O uso do computador com a linguagem de programação faz do aluno e de seu desenvolvimento intelectual, o foco da dinâmica educacional. Nessa abordagem, o profissional da educação perde o *status* exclusivo de repassador de conteúdos e atua, também, no sentido de ser o facilitador da aprendizagem. Isso porque, além de introduzir formalmente os comandos da linguagem LOGO, o professor será elemento fundamental de apoio ao aluno durante a atividade de programação, remindo dúvidas e propondo melhorias em seus trabalhos. Portanto, o LOGO não é somente uma linguagem de programação, mas uma maneira de se conceber e utilizar a programação de computadores (Barrella e Prado, 1996). Essa retomada às bases teóricas de Papert visou salientar não apenas o aspecto pedagógico do LOGO, mas também estabelecer argumentos, que justifiquem a diferenciação entre construtivismo e construcionismo.

CAPÍTULO IV

ALUNOS E PROFESSORES:
AGENTES DO CONHECIMENTO NA ABORDAGEM CONSTRUCIONISTA

As crianças, tal como todas as outras pessoas,
não preferem a facilidade, querem o desafio
e o interesse, o que implica dificuldade.
Seymour Papert

Neste capítulo busca-se formalizar as implicações diretas para os sujeitos ligados ao processo de ensino-aprendizagem usando linguagens de programação de computadores, mostrando que essa atividade é norteada por uma dinâmica reflexiva.

4.1 – Implicações do uso de computadores e linguagens de programação na docência

De forma geral, o uso de computadores na Educação prevê uma confluência entre as áreas da Informática e da Pedagogia. Entende-se, assim, que os profissionais envolvidos devem possuir ou desenvolver habilidades não apenas técnicas, mas também pedagógicas, as quais servirão de suporte para o exercício de uma prática reflexiva, na qual o computador torna-se ferramenta viabilizadora.

Zeichnner (1993) manifesta um estranhamento quanto ao uso do conceito de reflexão relacionado à prática pedagógica. O autor chega a tratar a falta de entendimento sobre o assunto como um *slogan* para as reformas de ensino e para a formação de professores e acaba por afirmar que "o termo reflexão perdeu virtualmente qualquer significado" (Zeichnner, *op. cit.*, p.15). Prado (1999), por sua vez, indica que a reflexão enquanto concepção de ensino e construção de conhecimento está associada ao pensamento de John Dewey, "que se referia à aquisição do saber como fruto da reconstrução da atividade humana a partir de um processo de reflexão sobre a experiência, continuamente repensada ou reconstruída" (Prado, *op. cit.*, p.49). Para

Schön (1992), finalmente, esse processo se constitui pelo intermédio da “reflexão-na-ação” e “reflexão-sobre-a-ação”.

> A reflexão-na-ação desenvolve-se simultaneamente com a ação (diálogo do pensamento com a situação-problema), quando o professor vai ao encontro ao aluno, procura compreender o seu processo de desenvolvimento (interação), ajuda-o a formalizar o seu “conhecimento em uso” – conhecimento intuitivo, espontâneo, experimental – e a articulá-lo com o conhecimento científico (intervenção). Durante esse processo, o professor pode “reformular suas ações no decurso de sua intervenção”, levantar e testar novas hipóteses “que demandam do professor uma forma de pensar mais flexível e aberta” e o faz construir novas teorias sobre o caso, além de vivenciar a dialética da aprendizagem. (PRADO, 1999, p.49).

Observa-se, portanto, que refletir na ação refere-se à elaboração mental que ocorre simultaneamente à ação do professor. Nota-se um movimento criativo no qual o professor é levado a reelaborar estratégias durante seu exercício prático. Esse dinamismo empregado no processo de ensino-aprendizagem se mostra contraditório em relação aos princípios da racionalidade tecnicista, afinal, a abertura recorrente ao professor reflexivo permite-lhe transcender a aplicação de regras ou métodos de ensino.

Já o exercício da “reflexão-sobre-a-ação” configura-se posteriormente à ação. O processo está ligado à atitude de buscar uma análise, ou ainda, uma reconstrução crítica das ações que constituem o fazer pedagógico do professor. Nesse momento de reflexão sobre seu labor, o professor busca a compreensão de sua própria atuação enquanto agente educacional.

> “A reflexão-na-ação, portanto, representa o saber fazer (que ultrapassa o fazer automatizado) e a reflexão-sobre-a-ação representa o saber compreender. São dois processos de pensamento distintos, que não acontecem ao mesmo tempo, mas que se completam na qualidade reflexiva do professor” (FREIRE e PRADO, 1995, p.235)

Zeichnner (1993) retoma de Dewey as atitudes inerentes à ação reflexiva: a abertura de espírito, a responsabilidade e a sinceridade. Por abertura de espírito compreende-se a atitude positiva de estar sempre disposto a avaliar alternativas, escutar críticas e ser suscetível ao debate e ao confronto de idéias. No que tange à responsabilidade, entende-se que a atitude do profissional reflexivo está compromissada com uma Educação sem finalidades imediatistas, ao contrário, no decorrer do tempo,

apresenta respostas a nível pessoal (aluno), social/político (comunidade) e acadêmicas (intelecto). Finalmente, a sinceridade tem a significação de engajamento profissional com um fazer pedagógico motivado, prazeroso e em contínua transformação.

Na dimensão deste trabalho, o uso do computador com linguagens de programação configura-se como um elemento central. A utilidade do computador é a de uma ferramenta que auxilia a instauração de uma prática reflexiva, que incita o "pensar-com" (por intermédio do computador – reflexão-na-ação) e também o "pensar-sobre" (pensar sobre o pensar, por intermédio do computador – reflexão-sobre-a-ação) não só para professores, mas também para os seus alunos (Prado, 1999; Almeida, 1999).

Como visto anteriormente, a atividade de programar um computador exige raciocínio lógico, reflexão, pesquisa e envolvimento de alunos e professores. Nessa dinâmica, o processo de DERD faz-se indispensável (seção 2.5). Na perspectiva discente, a reflexão-na-ação e a reflexão-sobre-a-ação permeia a atividade de programação. Nesse sentido, o LOGO (seção 3.2) oferece duas possibilidades de trabalho: o modo direto e o de edição de procedimentos. No primeiro, ao enviar um comando para a Tartaruga, o discente recebe um *feedback* imediato e passa a refletir sobre aquele passo (reflexão-na-ação) podendo aceitá-lo ou modificá-lo. Já no "modo de edição" de procedimentos, o aluno é levado a estruturar suas idéias no formato de programas de computador, ou seja, os comandos utilizados envolvem relações lógicas visando à resolução de uma situação problema. Dessa forma, ao criar um procedimento, o aluno tem que descrever suas idéias e só após isso, submetê-lo à execução no computador. O *feedback* continua a ocorrer, mas numa forma diferente, pois o aluno terá um retorno sobre a totalidade de seu programa e dessa forma será levado a refletir sobre um conjunto de comandos que formam o seu programa (reflexão-sobre-a-ação).

Ao tratar da função do professor no processo de ensino e aprendizagem com linguagens de programação destaca-se sua importância como facilitador. Na interação sujeito-aluno X computador, pode acontecer de o discente encontrar-se em uma situação na qual ele não sabe, por exemplo, a aplicação de um conceito ou o nome de um comando. Nesse ponto, a ação docente se efetiva, buscando, junto ao educando, sanar aquele momento de impasse, que interrompe o processo de DERD de idéias do

aluno. O profissional da educação reativa a dinâmica e ao fazê-lo, acontece também para ele a reflexão-na-ação e sobre-a-ação.

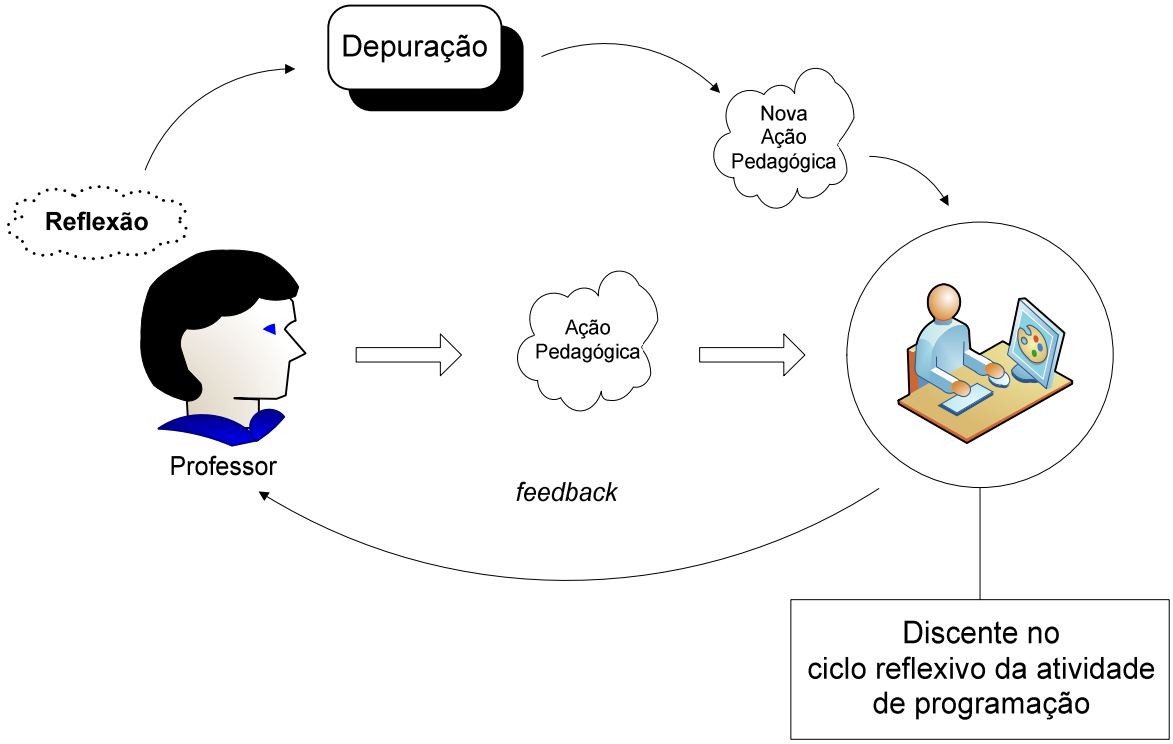

Figura 4.1 – Configuração da ação reflexiva do professor (Adaptado de Prado, 1999, p. 18.)

A Figura 4.1 apresenta a dinâmica de ação reflexiva do professor durante o processo de DERD do aluno. Configura-se para o professor, naquele processo, o momento de reflexão-na-ação e também a reflexão-sobre-a-ação. É durante atividade programação realizada pelo aluno que ocorre a ação pedagógica do professor. Como facilitador da aprendizagem, ele busca ampliar, junto ao educando, estratégias cognitivas de solução para o problema em questão (aprendizagem em conjunto). Nesse momento, configura-se para o professor a reflexão-na-ação, pois ele deve compreender as idéias estruturadas pelo aluno e interferir positivamente de forma a suprir as necessidades do aluno. Nota-se ainda, que após sua atuação pedagógica, o profissional poderá/deverá buscar colher impressões sobre sua intervenção enquanto facilitador da aprendizagem, reelaborá-la e ampliá-la. Esse fato constitui a reflexão-sobre-a-ação do próprio professor, reiterando-se aí a recorrência do processo reflexivo (Prado, 1999).

4.2 – Ação & Reflexão: indicativos para a atuação do professor

De maneira geral, o profissional que atua em informática na educação é um dos elementos fundamentais para o sucesso dos processos pedagógicos que envolvem o uso do computador como ferramenta. Disso deriva a importância de sua forma de atuação enquanto agente da aprendizagem em ambientes computacionais. Pensar nesse sentido leva a acreditar que parte desse esperado sucesso está relacionado, entre outros fatores, ao preparo para o exercício de sua profissão. Esta seção traça indicativos que devem ser inerentes à prática dos agentes educacionais que usam a informática na Educação.

Como dito anteriormente, o uso de computadores na educação se concretiza em uma confluência entre as áreas da informática e da pedagogia. Percebe-se que o profissional da área deve possuir ou desenvolver habilidades não apenas técnicas, mas também pedagógicas, as quais servirão de suporte para o exercício de uma prática reflexiva. Dessa maneira, de acordo com Valente (1993), o uso do computador no trabalho com alunos cria situações de conflito que levam o professor a questionar sua ação, refletir sobre sua prática pedagógica, refletir e questionar a prática pedagógica a que está submetido e a iniciar um processo de mudança de postura como educador, diferente daquela de professor repassador de conhecimento.

Todo esse quadro é desafiador para os professores, que vivenciam, em conjunto com seus alunos, um contínuo processo de aprendizagem seguindo um processo reflexivo. Portanto, é necessária a qualificação tanto técnica quanto pedagógica desses profissionais. As competências exigidas do professor “difusor do conhecimento”, não cabem nesse processo, que diz respeito a uma nova realidade:

> O professor torna-se um animador da inteligência coletiva dos grupos que estão ao seu encargo. Sua atividade será centrada no acompanhamento e na gestão das aprendizagens: o incitamento à troca dos saberes, a mediação relacional e simbólica, a pilotagem personalizada dos percursos de aprendizagem, etc. (LÉVY, 1999 p.171)

O pensamento de Lévy (1999) conforma-se ao de Almeida (1999) que nos chama a atenção para a maneira pela qual o professor mediador deve atuar no momento de suas intervenções junto a um aluno, a fim de “promover o pensamento do sujeito e

engajar-se com ele na implementação de seus projetos, compartilhando problemas, sem apontar soluções; respeitando os estilos de pensamento e interesses individuais; estimulando a formalização do processo empregado; ajudando assim o sujeito a entender, analisar, testar e corrigir os erros" (Lévy, *op. cit.*, p.29).

Ambos os autores mostram que o papel desse professor é o de provocar interações, o uso das ferramentas de construção do conhecimento, propor desafios e aprender junto de seus alunos. Em consonância, Altoé (1996) apregoa que nessa nova realidade, o professor passa a ser o facilitador ou o norteador do processo de aprendizagem e construção do conhecimento. Essa postura é conflitante com a do "professor entregador de conteúdos" e abre espaço para o questionamento crítico, o debate, o incentivo à pesquisa e à aprendizagem conjunto e contínua. É uma perspectiva compatível com a de Paulo Freire (1995, 1976) que apregoa uma atitude pedagógica que visa despertar a curiosidade, o questionamento, a investigação, a criatividade e um ambiente onde o professor "além de ensinar, [...] aprende; e o aluno, além de aprender, ensina" (Prado, 1999, p. 43).

Conceber o processo de formação de um professor nesses moldes é complexo e vem sendo objeto de estudos (Altoé, 2001; Almeida, 1999a; Almeida, 2000; Marchi, 2001; Prado, 1993; Prado,1998). Um fato balizador diz respeito à necessidade de se extrapolar o mero treinamento em uma determinada ferramenta (*software*), sinalizando a necessidade de um processo de formação, permeado pela prática e reflexão-sobre-a-prática.

Não existem, entretanto, cursos específicos para a formação de professores que irão atuar nesses processos educacionais. Os recursos humanos que são selecionados para exercer a profissão de professor dessas unidades curriculares são, geralmente, os bacharéis em Ciência da Computação ou em Sistemas de Informação, engenheiros, entre outros. Essas áreas são, todavia, essencialmente técnicas e que, em sua estrutura curricular não integram elementos pedagógicos visando à preparação para o possível exercício do magistério. Nesse sentido, é possível afirmar que o bacharel transforma-se em professor sem ter o devido aporte pedagógico para o exercício de sua profissão.

Mesmo após essa constatação, seria muito pretensiosa a afirmativa de que não existem agentes educacionais ligados ao processo de ensino usando linguagens de programação, que não conseguem realizar um trabalho pedagogicamente eficaz. Entretanto, não se pode deixar de assumir, que uma devida preparação/formação para o exercício da profissão docente, viria ampliar as possibilidades pedagógicas desses profissionais e cooperariam para minimizar a situação que é apontada por Petry (2005), a qual afirma que o aproveitamento alcançado nas unidades curriculares que envolvem programação é demasiadamente baixo e constitui-se como um elemento de preocupação para os profissionais ligados à atividade.

Não se pode estabelecer que a ausência de preparo para a docência seja a principal causa da situação que envolve o ensino de programação. Entretanto, é admissível que um embasamento que articulasse prática de ensino, a reflexão e conhecimentos teóricos sobre aspectos pedagógicos construcionistas poderiam sim, melhorar a ação pedagógica dos sujeitos envolvidos com o ensino de programação.

Uma contribuição deste trabalho é a de fornecer indicativos do perfil do profissional que está atuando ou pretende atuar dentro de uma perspectiva construcionista. Para isso, aliou-se a experiência pessoal do pesquisador aos trabalhos de Papert (1980/1985, 1994), Almeida (1996) e Ribeiro (1994) a fim de sintetizar algumas das características intrínsecas do educador que trabalha com ensino de programação de computadores em nível superior.

Fundamentalmente, esse profissional encara os novos dispositivos tecnológicos como instrumentos auxiliares na implantação de uma proposta educativa criativa e menos tecnicista. Nesse sentido, o professor procura o domínio das ferramentas computacionais as quais se propõe a utilizar junto com seus alunos, tirando delas o maior proveito possível em sua prática. Além disso, prioriza a expressão de idéias usando as linguagens de programação seguindo o processo de DERD, buscando promover um equilíbrio entre teoria e prática.

No exercício pedagógico revela-se um professor que abre espaços para o exercício de reflexão conjunta, valorizando o trabalho cooperativo e provocando situações de desafio para os educandos. É explorador de temas emergentes, contextualizados e que são significativos para os alunos. Assim, não menospreza a

bagagem cultural dos educandos e sua realidade, com as quais trabalha e busca aprimoramentos. Ao introduzir novos temas e conceitos promove conexões entre estes e os já existentes, procurando conhecer o desenvolvimento cognitivo de seus alunos visando uma mediação mais adequada. Encara os erros como "aproximações do resultado esperado e não como fracasso ou incompetência" (Almeida, 1999, p.44).

O profissional que se baseia no construcionismo, considera o processo de aprendizagem não exclusivo dos discentes, encarando a docência como uma oportunidade de aprender em conjunto com seus alunos. Assim, mediante o surgimento de problemas, tem compromisso com a pesquisa, levanta hipóteses, realiza experimentações, reflexões e depurações com o objetivo de melhorar e validar sua prática (Almeida, 1999). Atua explorando e divulgando referências bibliográficas, documentos eletrônicos, sítios da Internet como complemento ao seu fazer pedagógico.

Finalmente, é um professor engajado, dedicado e que tem gosto pelo fazer pedagógico, mostrando-se defensor do *software* livre como instrumento de inclusão digital e fomentador do processo educativo.

4.3 – Alunos e a aprendizagem de programação de computadores

O foco da abordagem construcionista é o aluno e a sua aprendizagem. Petry (2005) e Martins e Correia (2003) mostram, em linhas gerais, que a dinâmica de ensino-aprendizagem de programação e unidades curriculares correlatas tem sido bastante improdutiva, com grande índice de reprovação e desistência, pois os alunos apresentam grande dificuldade em expressarem suas idéias, de forma lógica, usando linguagens de programação. Em conformidade, Pereira *et al* (2004), Rodrigues Júnior (2002), Schultz (2003), Delgado *et al* (2004) e Chaves de Castro *et al* (2004) mostram que esses fatos contribuem para a desmotivação, apatia e baixa auto-estima dos educandos.

A complexidade que permeia o processo de ensino-aprendizagem de programação de computadores envolve as ferramentas de programação (linguagens), o "sujeito professor" e sua metodologia de condução das unidades curriculares, o "sujeito aluno" e, evidentemente, as dificuldades inerentes à área. Até então, já foram situados dentro da dinâmica educacional, todos esses "elementos", com exceção do "sujeito aluno".

Construir o conhecimento através de linguagens de programação não é uma tarefa simplória, ligada à simples cópia, replicação de informações adquiridas dos professores/livros, sendo fundamental situar o campo de ação do educando. O aluno precisa estar ciente de que está inserido em uma dinâmica onde ele não é mero receptor de conteúdos ou, em outras palavras, um espectador. Pelo contrário, ele deve se envolver e se interessar pelas as ferramentas de *hardware e software*, com as propostas de solução de problemas (que serão implementadas sob a forma de programas de computador), buscando ir além da compreensão de conceitos isolados.

Maltempi e Valente (2000) sugerem que o desenvolvimento cognitivo ligado à atividade de programação é um processo ativo de construção e reconstrução das estruturas mentais, onde o conhecimento não pode ser transmitido simplesmente do professor para o aluno. Dessa maneira, defendem que os alunos devem colocar as mãos à obra (*hands-on*) no desenvolvimento de atividades que privilegiam o processo de DERD, ao invés de serem contempladores da fala de seus tutores. Quer dizer, ao programar um computador, um aluno deve exercer o ciclo DERD a fim de valorizar tal atividade em termos de aprendizagem. Não vale, nesse sentido, o "fazer por fazer", ou a simples digitação (cópia) de um programa no computador sem a efetiva reflexão e depuração.

Fundamental é que o educando tenha identidade com o seu fazer (*head-in*). Em outras palavras, o educando deve se entregar àquilo que se propõe a fazer, não por obrigação, mas por inteira afinidade. Afirma-se isto porque, apesar da alta carga de dedicação, concentração e disciplina que a atividade de programação de computadores exige, o objeto final construído traz ao seu construtor uma satisfação gratificante e prazerosa. É o que Valente (1999) chama de *empowerment*: a sensação que é experimentada pelos aprendizes ao conseguir elaborar um "produto que eles não só construíram, mas compreenderam como foi realizado". Dessa forma, os aprendizes "podem falar sobre o que fizeram e mostrar esse produto para outras pessoas [...]" e ao fazê-lo acabam por se permitirem "uma grande massagem no ego" (Valente, *op. cit*., p.82).

Percebe-se, portanto, que um fator decisivo para o sucesso do educando é a sua auto-estima. Essa peculiaridade assegura-lhe a perseverança e a motivação para

seguir na aprendizagem. Tais circunstâncias indicam sentidos do que está fazendo, porque está fazendo e onde vai chegar ao final do processo.

O educando, principalmente o vinculado ao Nível Superior de ensino, possui a responsabilidade de situar o seu grau de aprendizagem e de (re)agir, ponderando seu nível de envolvimento com a construção de seu conhecimento. Dessa maneira, o aluno acaba por desenvolver a convicção de que ele é o principal responsável pela sua formação podendo gerenciar sua aprendizagem.

Nesse sentido, Papert (1993/2008) também traz colaborações. Como foi dito no início dessa sessão, o foco da proposta construcionista diz respeito ao aluno e sua aprendizagem. Entretanto, na visão tradicional da Educação o controle da aprendizagem está nas mãos do professor e o foco do processo é o ensino. Papert (1993/2008) questiona a ideologia hierárquica escolar, que coloca o ensino como o processo ativo, e não a aprendizagem: "O professor está no comando e é, portanto, quem precisa de competência; o aprendiz tem apenas que obedecer a instruções" (Papert, *op. cit.*, p.88). As indagações de Papert (1993/2008) refletem sobre o termo "didática" – sempre ligado à competência do professor durante seu exercício profissional – e destaca a importância de que o educando deve ser igualmente competente durante o seu processo de aprendizagem.

Segundo o Dicionário Eletrônico Houaiss da Língua Portuguesa (2001), o temo "didática" é assumido como "arte de transmitir conhecimentos" ou ainda "arte de ensinar". Em Ferreira (2004) encontra-se o mesmo termo como sendo a "técnica de dirigir e orientar a aprendizagem" ou ainda "técnica de ensino".

> E quanto aos métodos para aprender? Que disciplinas são oferecidas aos que desejam tornar-se aprendizes competentes? [...] Não há quaisquer designações semelhantes para áreas acadêmicas que apóiem a arte de aprender. [...] Sob diversos nomes, a Pedagogia como arte de ensinar foi adotada pelo mundo acadêmico como uma área respeitável e importante. A arte de aprender é uma órfã acadêmica. (PAPERT, 1993/2008, p.87)

Papert (1993/2008) protesta quanto à exaltação do ensino em detrimento da aprendizagem. Em sua defesa, propõe o termo "matética[9]" para definir a "arte de aprender" ou ainda "o conjunto de princípios norteadores que regem o aprendizado" (Papert, 1980/1985, p. 74). O autor suscita o termo "heurística" explicando que ele se difere de "matética", uma vez que vem ele sendo utilizado como conjunto de regras e métodos aplicáveis à resolução de problemas. E acrescenta: "a matética é para a aprendizagem o que a heurística é para a solução de problemas" (Papert, 1993/2008, p.89). Nesse sentido, Altoé e Penati (2005) explicam que

> utilizar uma "regra" para resolver problemas [heurística] não promove a aprendizagem, mas sim o pensar sobre o problema proposto. A "arte de aprender" [matética] seria, então, em vez de pensar em regras eficazes para aplicação imediata, procurar explicações de como trabalhar com essas regras, contribuindo, em longo prazo, para uma aprendizagem efetiva. (ALTOÉ e PENATI, *op. cit.*, p.9)

Papert (1993/2008) reconhece o valor da heurística e não despreza sua aplicação. Lembra que essa idéia remonta a Descartes, rememorando, contemporaneamente, o matemático húngaro George Polya, autor do livro "*How to solve it*". Entretanto, Papert (1993/2008) acredita que a aplicação direta de regras na solução de problemas acomete os estudantes na busca de, ao iniciarem a solução, o fazê-lo com muita "pressa de terminarem e iniciarem o seguinte" (Papert, op. cit., p. 90). Nesse sentido inicia uma "perspectiva matética" para a aprendizagem na qual deve ser valorizado: o "dar-se tempo", ou seja, o refletir não tensionado sobre um problema em questão sem retalhar o tempo é o que promove a aprendizagem e a possibilidade de ao compreender a situação, resolvê-la, construir o conhecimento e aplicá-lo novamente em situações similares; a discussão em grupo, princípio matético central, que busca valorizar o falar e a interação entre alunos e professores, levando os aprendizes a romper a barreira do medo de expor suas dúvidas e apresentarem o que pensam a respeito do que está sendo tratado; o estabelecer de conexões, que pode ser também entendido como a valorização de associações fortemente significativas/evocativas entre um novo conhecimento e o que já se interiorizou.

[9] Matética, segundo Papert (1994), derivaria do grego *mathematikos* – "disposto a aprender" – ou ainda do verbo *manthanei* – aprender.

Assim, o discente de um curso em Nível Superior – frente à proposta construcionista – assume uma parcela de responsabilidade no processo de aprendizagem, o que acaba por revelar uma atitude "mateticamente" madura. Seu engajamento matético aumentará sua auto-estima, o mostrará como um sujeito motivado, com facilidade em superar obstáculos, aprender em conjunto e o principal: possuidor da habilidade de aprender.

Os trabalhos de Rocha (1988), Maltempi e Valente (2000) e Martins e Correia (2003), relatam estudos assemelhados ao presente trabalho, abrindo espaço para comparações, complementações e, em sentido geral, ampliação da compreensão do tema em questão.

Rocha (1988) descreve uma experiência realizada na UNICAMP, no curso de Bacharelado em Ciência da Computação, com duas turmas de primeiro período, na qual se buscou introduzir o LOGO como linguagem de programação inicial, visando uma conexão posterior com a linguagem PASCAL, tradicionalmente explorada durante os três primeiros períodos do curso. O estudo relata o preconceito que associa a linguagem LOGO como sendo, exclusivamente, de caráter infantil e, portanto, inadequada ao uso acadêmico. Segundo a autora, a primeira barreira a ser vencida foi a de convencer os responsáveis pelo Curso da validade da linguagem, além de ter sido necessário assegurar que os alunos sairiam do curso com a mesma bagagem de PASCAL que, comumente, era explorada. Rocha (1988) explicita que o objetivo daquele trabalho era o de comprovar que o LOGO poderia ser trabalhado com adultos, especialmente, os que pretendiam se tornar especialistas em computação. O curso ministrado foi dividido em duas partes, sendo que a primeira foi dedicada à introdução aos conceitos de programação usando o LOGO e a segunda, estabeleceria uma conexão dos conceitos usando a linguagem PASCAL. A pesquisadora relata que os resultados alcançados com o LOGO foram excelentes, destacando-se o nível de estruturação dos programas, uso de técnicas como recursão e passagem de parâmetros. Entretanto, relata que a transição para o PASCAL foi feita com menos facilidade do que o esperado e que os programas escritos em PASCAL foram menos sofisticados que os elaborados em LOGO. A autora atribui o ocorrido à mudança de ambiente e ao tipo de problema que passou a ser resolvido: com o LOGO, em sua maioria, eram problemas essencialmente gráficos, já com o PASCAL eram de

natureza numérica e literal. O curso foi avaliado pelos alunos como desestimulador, pois segundo eles o LOGO não era, de forma alguma, útil para suas formações acadêmicas. Além disso, afirmaram que não tinham aproveitado o modo como o PASCAL foi trabalhado. Em sentido oposto ao das avaliações dos alunos, Rocha (1988) relata que o professor que assumiu a turma em seu segundo semestre afirmou ter recebido uma turma com muita base e que conseguiu avançar em sua disciplina com eficácia superior, chegando a cobrir parte da matéria do curso seguinte ao seu (3º período). Em suas conclusões, a autora confirma a hipótese de que existe uma idéia pré-concebida em torno do LOGO, que diz respeito ao seu uso exclusivamente infantil, e mais, afirma que iniciativas precisam ser desenvolvidas a fim de se mudar tal concepção. Propõem que os pesquisadores em LOGO avancem no sentido de se explorar não apenas os aspectos superficiais da linguagem, que ultrapassem seus aspectos gráficos e que aprofundem a complexidade dos problemas que podem ser resolvidos com o LOGO.

Maltempi e Valente (2000) construíram uma revisão bibliográfica no qual a atividade de programação de computadores é apresentada como um processo rico e favorável à construção de conhecimentos, fornecendo indicativos de como promovê-las em ambientes de ensino-aprendizagem. Os autores destacam em seu estudo que o sistema educacional privilegia atividades que contemplam a memorização e reprodução de informações, além de coibir e encarar o erro de forma punitiva. Afirmam que esse modelo de ensino inibe e inviabiliza o pensamento crítico dos educandos, apresentando a atividade de programação como uma tarefa que contraria essa forma de transmissão de conhecimento, constituindo-se fomentadora de uma postura ativa e criativa do aluno. Dessa forma, mostram que parte dos problemas envolvidos no processo de ensino-aprendizagem de programação advém do choque entre o modelo tradicional de ensino com o que é exigido e esperado dentro das unidades curriculares de programação de computadores. Ao descreverem a atividade cognitiva de programação de computadores enfocam o processo de DERD e destacam que a fase de depuração é a que prove efetivamente a revisão de conceitos, estratégias, erros e leva os sujeitos da aprendizagem a efetivamente construírem seu conhecimento. Mostram que o processo de ensino-aprendizagem é, para alunos e professores, uma atividade de cunho reflexivo que desenvolve o pensamento crítico e uma constante revisão de conceitos. Maltempi e

Valente (2000) indicam que melhores resultados podem ser alcançados nas unidades curriculares de programação quando são utilizadas atividades de interesse dos alunos e que proporcionam seus engajamentos (*hands-on/head-in*), e sugerem a abordagem pedagógica baseada em projetos para implementar tais atividades.

Martins e Correia (2003) desenvolveram um experimento similar ao que foi empreendido neste trabalho. Os autores descrevem uma experiência realizada com alunos ingressantes do curso de Ciência da Computação das Faculdades Integradas de Rondonópolis (FAIR), no qual foi oferecido um curso livre da Linguagem LOGO, na modalidade de extensão universitária, visando fortalecer e apoiar o desenvolvimento do raciocínio lógico dos participantes. Diferentemente, no presente trabalho, a análise se dirigiu a um grupo de alunos do último período de um curso de Sistemas de Informação. Para Martins e Correia (2003) o objetivo específico do trabalho desenvolvido foi o de observar da reação dos alunos ao utilizarem o LOGO. Como objetivo secundário, destacaram a viabilidade do LOGO concretizar um projeto interdisciplinar entre as unidades curriculares de matemática básica e algoritmos. Os autores relatam que o experimento foi realizado com um grupo de alunos que variou de cinco a vinte e um presentes, nas oito horas de atividades planejadas para cinco encontros. Foi utilizado o *software* SuperLogo como ambiente de programação, com o qual se desenvolveram atividades geométricas buscando reforçar a capacidade de abstração para a elaboração de algoritmos, coordenação cartesiana, uso de estruturas de controle e escrita de procedimentos. Os autores consideram que os resultados alcançados foram satisfatórios, concluindo que o LOGO é uma ferramenta útil para promover a interdisciplinaridade entre as unidades curriculares envolvidas. Destacam ainda, que embora o curso tenha sido finalizado com apenas cinco alunos presentes, foi possível notar mudanças comportamentais nos participantes. Entre elas: atitude proativa dos alunos em relação às atividades regularmente oferecidas, melhor compreensão do processo de resolução de problemas no computador, minimização de dificuldades com relação aos conceitos básicos da teoria de programação de computadores, além da empatia com o ambiente de programação SuperLogo. Martins e Correia (2003) apontam contribuições que justificam a problemática envolvida no ensino de programação, sintetizando-as: nas deficiências de formação no ensino médio e fundamental, no despreparo e imaturidade dos egressos no

Ensino Superior; na incompatibilidade de perfil dos alunos para a área da computação e no formato dos processos seletivos desses universitários. Ponderam ainda que o ensino de programação não é uma atividade simplória, devendo os profissionais desta modalidade buscar uma melhor adequação para o exercício de sua atividade docente.

CAPÍTULO V

LOGO: PENSO E EXISTO

Em uma piada clássica, uma criança fica depois da aula para fazer a pergunta: "Professora, o que eu aprendi hoje?". A professora, com ar de surpresa, questiona: "Por que está perguntando isso?". A criança responde: "Papai sempre me pergunta e eu nunca sei o que dizer!".
Seymour Papert

Este capítulo apresenta fragmentos de falas de alunos e professores, levantados durante a pesquisa de campo, acompanhados de sua discussão. O trabalho desenvolvido é de natureza qualitativa e exploratória, tendo sido utilizados o questionário e a entrevista semi-estruturada como instrumentos de coleta de dados (Anexos 1 e 2).

Lakatos e Marconi (2001) definem o questionário com um meio de coleta de dados caracterizado por apresentar uma série ordenada de perguntas. A entrevista semi-estruturada é aquela norteada por aspectos pré-definidos e permite ao entrevistado a oportunidade de expressar sua opinião e percepções. Bogdan e Biklen (1994, p.193) confirmam essa característica ao afirmarem que as entrevistas semi-estruturadas possibilitam "oportunidade de compreender como é que os próprios sujeitos estruturam o tópico em questão".

Os questionários foram utilizados com a finalidade de traçar o perfil de uma turma de formandos do curso de Sistemas de Informação da "Universidade Beta" e suas relações com as unidades curriculares de programação de computadores. Esse instrumento serviu como base de planejamento do trabalho desenvolvido utilizando a linguagem de programação LOGO em uma unidade curricular de 40 horas. As respostas foram categorizadas e interpretadas à luz de uma análise de freqüência representada por gráficos.

Inicialmente, foram convidados seis alunos do grupo para participarem das entrevistas. Entretanto, apenas cinco se dispuseram a participar do processo. O critério

utilizado para a seleção foi o comprometimento dos alunos com as questões referentes ao curso oferecido, bem como em relação à programação de computadores. Especificamente, esses alunos mostraram-se interessados, atentos, participativos e comprometidos com a aprendizagem em curso. A fim de ampliar a compreensão da dinâmica envolvida nas unidades curriculares de programação, foram entrevistados dois professores da área, docentes da "Universidade Beta".

As entrevistas foram gravadas, transcritas e submetidas à análise de conteúdo, de acordo com Bardin (1977). Os fragmentos de interesse foram identificados e categorizados, com base nas unidades de significado presentes nas questões de pesquisa: o processo de ensino-aprendizagem de programação, a relevância das unidades curriculares de programação, fatores motivacionais, a linguagem LOGO, a preparação para a docência e os papéis de alunos e professores.

A primeira etapa do trabalho dedicou-se à determinação do perfil de uma turma de 21 formandos do curso de Sistemas de Informação da "Universidade Beta" por intermédio da aplicação de um questionário. Nesse instrumento de coleta de dados enfocou-se, primordialmente, a relação pregressa da turma com as unidades curriculares de programação de computadores.

Em um segundo momento, os alunos cumpriram uma carga horária de 40 horas em uma unidade curricular denominada "Tópicos Especiais", a qual faz parte da estrutura curricular do curso de Sistemas de Informação. A condução da referida unidade curricular ficou ao cargo do pesquisador, sendo utilizada a ferramenta SuperLogo como ambiente de programação. Este *software* foi escolhido por diversas características, destacando-se a sua natureza livre, já estar regionalizado e amigabilidade com o usuário. Além disso, o laboratório de informática em que o curso foi desenvolvido possuía licenciamento da plataforma *Microsoft*, o que exigia um ambiente de programação compatível.

Durante a carga horária foram explorados e revistos vários conceitos ligados à programação de computadores, tais como: passagem de parâmetros, repetições, recursividade etc. Pedagogicamente, procurou-se privilegiar as características intrínsecas a um ambiente de aprendizagem construcionista. Além disso, tópicos ligados ao uso da informática na Educação foram explorados objetivando o surgimento de

questionamentos, debates e a introdução de novos conceitos. O curso foi conduzido de forma a propiciar atividades práticas que favorecessem o uso do computador com o SuperLogo. Os alunos tiveram oportunidades de explorarem a ferramenta de *software*, rever e, em alguns casos, aprender conceitos de programação. O objetivo desta etapa era o de estimular o exercício da atividade cognitiva de programação com o SuperLogo e explorar suas características.

Finalmente, após o término da unidade curricular de 40h, foram aplicadas as entrevistas junto a alunos participantes e professores de programação, visando à coleta de dados que favorecessem o entendimento das questões propostas na introdução deste trabalho. Esta parte da pesquisa de campo foi realizada fora do ambiente universitário.

5.1 – O perfil da turma pesquisada

A fim de se buscar um planejamento eficiente para o curso ministrado, foi aplicado um questionário no primeiro encontro com os alunos, o qual buscava fornecer elementos relacionados a suas práticas com linguagens de programação. O que se pôde constatar foi uma relação de aversão, desmotivação e de pouco aproveitamento no que diz respeito a tal área de conhecimento. As respostas das questões foram tabuladas e posteriormente sintetizadas no formato de gráficos que revelam o perfil da turma.

A Figura 5.1 mostra que cerca de um terço dos integrantes da turma considerou-se com boa proficiência na área de programação e nenhum se enquadrou na categoria “excelente”. Pouco menos de um terço (9%) posicionou-se em uma proficiência mediana. O terço restante resulta da soma de sofrível (29%) e péssimo (5%).

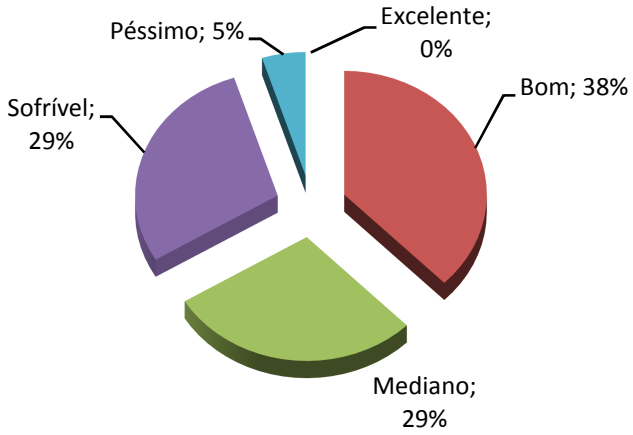

Figura 5.1 - Conhecimento adquirido em programação.

Apesar do que foi evidenciado anteriormente, a Figura 5.2 nos mostra que os discentes reconhecem a importância das unidades curriculares (UC) relacionadas à programação. Nenhum dos 21 integrantes as considerou “dispensáveis” ou mesmo “de baixa importância” o que nos leva a pensar que, embora tenham obtido um baixo aproveitamento nas referidas UC, esses carregam consigo a certeza de que sua falta de domínio na área lhes será prejudicial.

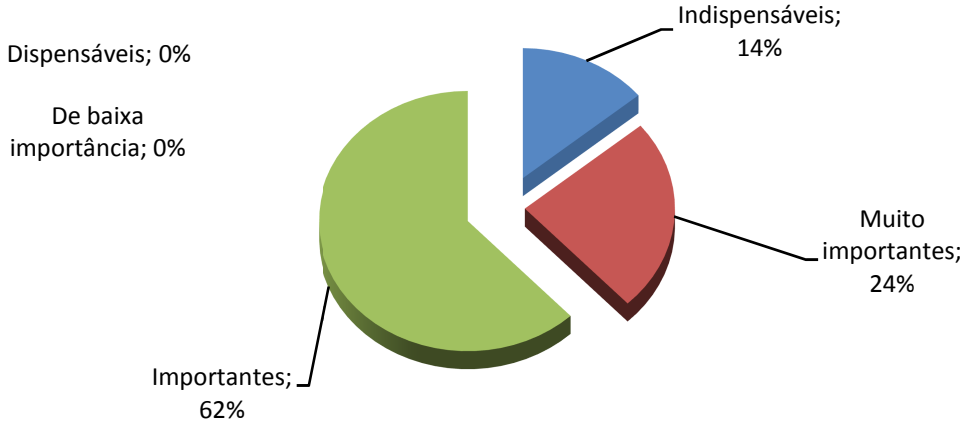

Figura 5.2 – A importância das UC de programação.

Um fato importante e que caracteriza de forma geral a turma pesquisada diz respeito a sua motivação durante os cursos de programação. Percebe-se que 66% da turma se declarou “desmotivada”, o que em um cruzamento com os dados da Figura 5.1, amplia o que até agora conseguiu ser evidenciado em termos de aproveitamento nas UC em questão.

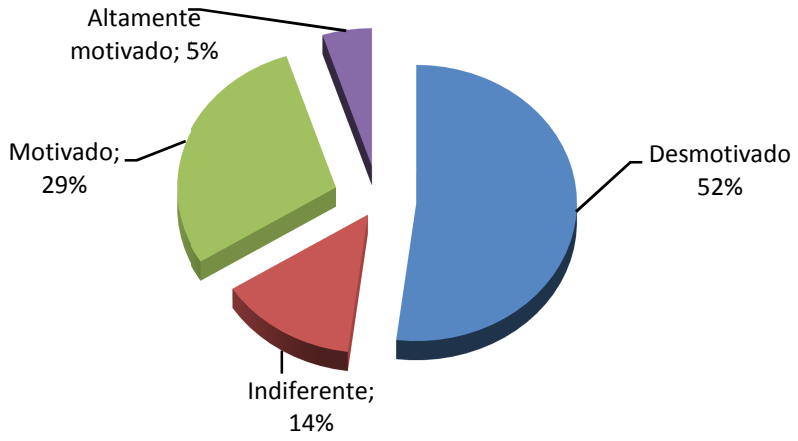

Figura 5.3 – Motivação dos alunos durante os cursos de programação.

A fim de se buscar o entendimento do grau de motivação da turma em relação às UC de programação, indagaram-se os motivos que influenciariam tal comportamento, deixando aberta ao aluno a opção de escolher por mais de uma entre as alternativas apresentadas: (A) Ao professor; (B) A estar fazendo o que eu realmente gosto; (C) Às ferramentas de *hardware* e *software*; (D) Outra opção. O aluno podia, ainda, sugerir uma resposta de cunho pessoal. Optou-se por dividir a análise dessa questão conforme:

Tabela 5.1 – Ocorrências de respostas referentes ao grau de motivação

Alternativa	A	B	C	D	A,B,C	A,C	A,D
Nº de alunos optantes	8	5	0	1	1	2	1

Percebe-se que a maioria dos alunos associou a motivação nas disciplinas à figura do professor, ou seja, existia uma expectativa sobre um trabalho diferenciado do mesmo com as UC de programação. Não se pode ignorar que cinco alunos atribuíram a si mesmos a obrigação de estarem motivados, provavelmente por terem consciência de que estavam fazendo um curso com o qual se identificavam. Poucos alunos associaram múltiplas alternativas, demonstrando a tendência de justificar a falta de motivação a um fator isolado. As ferramentas de *software* mostraram-se praticamente irrelevantes relacionadas ao quesito motivação.

Entre as opções abertas, a falta de prática em laboratório de informática e a má gestão administrativa do ambiente universitário, tiveram uma ocorrência cada. A opção aberta no questionário foi mal utilizada ou mal entendida, pois em sua maioria foi utilizada como uma justificativa para alguma das alternativas anteriormente selecionadas. Optou-se, portanto, por eliminar tais respostas.

Essa expectativa sobre o trabalho do professor de programação foi confirmada em uma questão que buscava ponderar o quanto o aluno considerava importante o envolvimento do professor com aquelas disciplinas. Dentro da questão proposta havia um espaço para uma breve justificativa sobre a opção escolhida. Nesta, pôde-se perceber que o professor além de dominar o conteúdo programático das UC, deveria ter a capacidade de transmiti-lo; além disso, uma relação prática com área (vivência de mercado) era desejada. A figura 5.4 demonstra o resultado alcançado, que se comparado com a questão anterior, confirma as suposições referentes ao trabalho diferenciado do professor de programação na visão daqueles alunos.

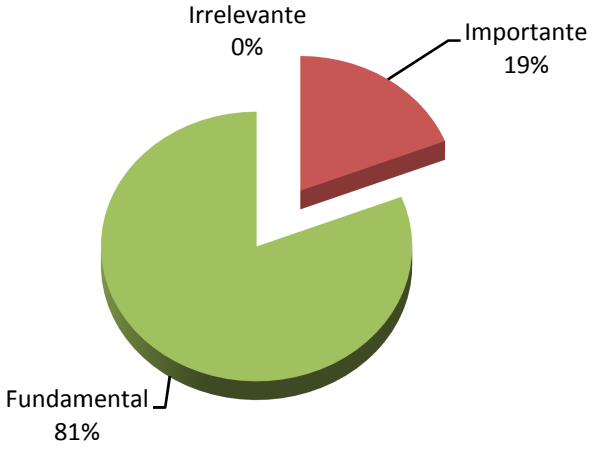

Figura 5.4 – A importância atribuída ao professor de cursos de programação.

Buscou-se, ainda, conhecer melhor o grau de responsabilidade do próprio aluno no processo de aprendizagem de programação. Dessa forma, perguntou-se a respeito da importância do envolvimento pessoal para se aprender programação. Os alunos reconheceram que para se obter êxito nas UC relativas à programação deveriam ter uma participação ativa. Entende-se, portanto, que o os sujeitos envolvidos no ensino-

aprendizagem de programação têm responsabilidades definidas e divididas. A figura 5.5 mostra as porcentagens alcançadas entre as três opções disponibilizadas na questão.

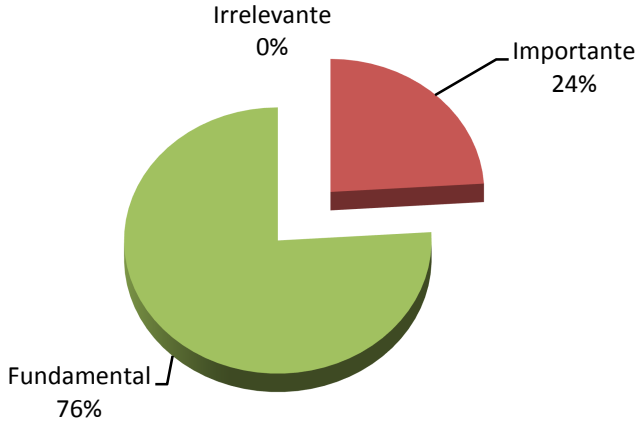

Figura 5.5 – A importância do envolvimento do aluno durante os cursos de programação.

A fim de proporcionar uma comparação entre o nível de dedicação dos alunos nos cursos de programação e o respectivo rendimento alcançado, foram propostas questões relativas a estes tópicos chegando aos seguintes indicadores:

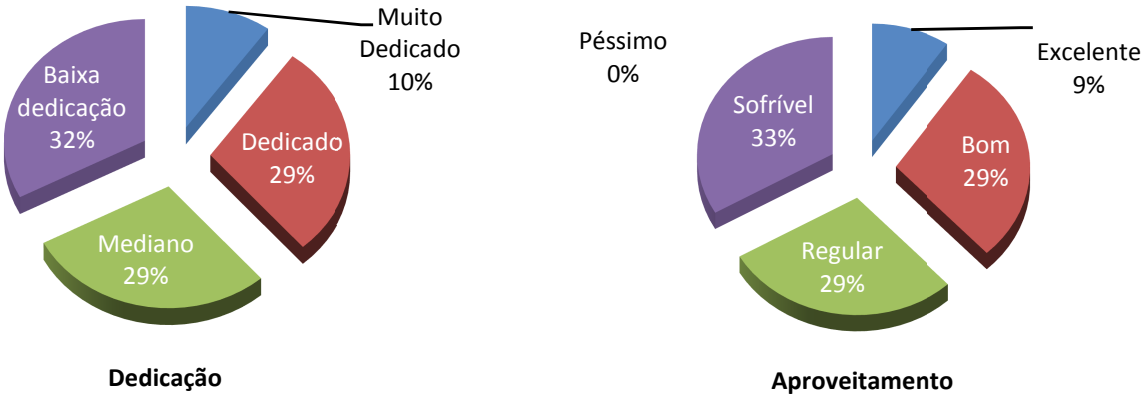

Figura 5.6 – Comparação entre dedicação e aproveitamento dos alunos em UC de programação.

O cruzamento desses dados sugere sinceridade dos participantes da pesquisa, pois em uma relação direta entre esforço pessoal e rendimento alcançado, os gráficos apresentam-se de forma praticamente idêntica. Da mesma forma, ao se

considerar a baixa e média dedicação, foi alcançado o indicador de 61%, que se cruzado com o fator motivacional, justifica os 66% dos alunos pouco ou nada motivados.

O questionário aplicado ainda possuía questões relacionadas às expectativas para o curso vindouro. Percebeu-se que os alunos tinham a esperança de conseguirem rever ou mesmo aprender, conceitos da programação em uma nova perspectiva. Eles acreditavam em uma metodologia de trabalho diferente das já experimentadas, relacionada a um novo professor com o qual, até então, não tinham tido a oportunidade de experimentar a programação. Para reforçar estas impressões, apresentam-se abaixo alguns trechos colhidos nas questões.

> **Aluno X:** *"Bem, como acredito que é uma junção de professor, aluno e ferramentas, nesse período ´estamos bem' para um aproveitamento quase total. Por que? Isso porque os alunos estão amadurecidos, a faculdade melhor equipada e temos um professor que veste a camisa e gosta do que faz".*
>
> **Aluno Y***: "Aprender o máximo que a disciplina tem a oferecer, tendo interação aluno/professor".*
>
> **Aluno Z:** *"Espero que o professor consiga quebrar esse medo, desmotivação da turma com relação à programação".*
>
> **Aluno W:** *"Espero que seja transmitido o conteúdo de forma compreensível, criativa e bem elaborada para que todos compreendam o intuito da matéria".*

A síntese do perfil dos alunos envolvidos na UC em questão destaca um grupo que, ao longo de sua formação acadêmica, não aproveitou satisfatoriamente os conteúdos relacionados à programação de computadores. A relação com tais UC remonta a um sentimento de desmotivação, frustração, descontentamento e de pouca interação entre professor, aluno e ferramentas de *software*.

5.2 – Análise dos dados: as falas dos alunos

Esta seção traz considerações que buscam estabelecer um panorama sobre a visão dos alunos sobre o processo de ensino-aprendizagem de programação de computadores, seguidas pela reflexão sobre a motivação dos alunos. Finalmente, apresentam-se e se discutem as impressões a respeito do uso da linguagem LOGO durante o curso de 40 horas, destacando-se as melhorias percebidas na compreensão de programação. Esta análise refere-se às questões de pesquisa A e B, norteadoras deste

trabalho: Quais os fatores que levam a motivação/desmotivação dos alunos no processo de construção do conhecimento pelo computador via linguagem de programação? O uso da linguagem LOGO aliada às tradicionalmente utilizadas (PASCAL, C, JAVA etc.) nas unidades curriculares de programação pode favorecer o processo de ensino-aprendizagem?

5.2.1 – A visão dos alunos sobre o ensino-aprendizagem de programação

Na perspectiva construcionista, a atividade de programação aparece de modo alternativo ao modelo de transmissão de conhecimentos, freqüentemente vivenciado em ambientes escolares dos mais diversos níveis de ensino. Trabalhar com programação de computadores em um ambiente de aprendizagem construcionista contraria o modelo transmissivo, exigindo um repensar sobre o lugar dos sujeitos no processo de ensino-aprendizagem. A aprendizagem de programação extrapola a repetição, a cópia de conteúdos e vem ao encontro a uma participação ativa, cooperativa e que exige dedicação dos praticantes. Programar um computador envolve o conhecimento de uma linguagem de programação, a capacidade de abstração sobre a realidade e, primordialmente, a expressão de idéias de forma lógica.

Ocorre que o ensino de programação de computadores enfrenta sérios problemas em relação ao rendimento alcançado por parte dos aprendizes. Esse fato coloca os sujeitos envolvidos na dinâmica educacional em uma situação de conflito entre o que é esperado em termos de construção de conhecimento e o que é realmente efetivado. Sobre isso, o Aluno4 diz:

> **Aluno4:** *No 3 e 4º período aconteceram muitos problemas, 90% da turma não estava satisfeita com a matéria de programação. [...] e eu não fiquei satisfeito, mesmo hoje... Bacharel em Sistemas de Informação eu acho que não tenho competência de tocar essa função [...]*

Dois aspectos particulares da questão pedagógica são apontados pelos Alunos 1, 5 e 2: a falta de introduções didatizadas[10] aos temas e a fragmentação do conteúdo, como numa linha de produção fordista:

[10] A idéia de didatização aqui utilizada corresponde a "transposição didática", no sentido proposto por Yves Chevallard (1991), a qual compreende uma dinâmica de transformações/adaptações dos saberes de referência, com o objetivo de torná-los mais acessíveis e apropriados ao processo educacional. Tal

> **Aluno1***: "O primeiro momento que a gente teve contato com a programação, foi muito brutal, assim... Porque como você nunca teve contato... é difícil. É... sem uma preparação primeiro, falar mais ou menos: você precisa programar para quê? [...] E eu acho que as matérias ficaram faltando mesmo, pelo fato disso: de uma introdução menos agressiva e mais produtiva".*

> **Aluna5:** *Programação é lógica. E eu acho que muita gente não está acostumada a raciocinar com lógica. [...] eu acho que a primeira dificuldade foi isso: a forma pela qual foi passado [...] eu acho que teve uma deficiência de didática* [do professor].

> **Aluna2:** [...] *igual a uma charge que eu tinha visto: tinha um cara que fabricava parafuso, e aí ele passava esse parafuso para uma outra coisa, que seria feita com o parafuso. E aí um dia ele falou: "que será que fazem com esse parafuso?" E aí lá no final, tinha lá, um equipamento. Então a pessoa está fazendo aquilo ali, mas não sabe para quê que serve. Como que é? O quê que vai ser? O quê que o parafuso vai fazer? Vai fazer parte de quê? É de alguma coisa... mas o quê que é?*

A Aluna3, por sua vez, retoma a questão da abordagem pedagógica, referindo-se a momentos de aprofundamento nos conteúdos:

> **Aluna3:** *No início eu achava a coisa mais linda do mundo. Quando você conseguia fazer o negócio era ótimo. Mas aí começou a complicar, complicar... foi ficando difícil, a turma inteira começou a desesperar [...] Virou um bicho de sete cabeças. [...] Foi muito difícil para mim.*

Em suas falas, os alunos explicitam seu descontentamento com a abordagem pedagógica utilizada no processo de ensino-aprendizagem de programação de computadores. Entendeu-se que as referidas UCs exigem uma nova forma de se proceder com vistas à aprendizagem: uma dinâmica que extrapole a transmissão de conceitos, voltando-se a uma prática criativa, situada, efetiva, com sentido de ser e identificada por objetivos claramente estabelecidos e inter-relacionados. A fala da Aluna3, em especial, parece evidenciar o distanciamento dos discursos docentes da zona de desenvolvimento proximal em que os alunos, pelo menos em maioria, se situam.

Apesar de tudo isso, os alunos consideram a programação de computadores como algo muito importante em sua trajetória acadêmica e profissional. Quando questionados em relação à relevância das UC de programação, os alunos foram muito claros em relação ao quanto seria importante o conhecimento da área:

> **Aluno1:** *O contato que você tem com a linguagem de programação é essencial para tudo o quanto é área de tecnologia que você vai estar no meio. Porque a linguagem de programação está*

---

transposição implica, em termos operacionais, dentre outras coisas, a complementação entre textos e figuras e entre conceituações e exemplos, a alternância de vozes entre professor e alunos, a apresentação de exercícios.

estruturada em tudo em quanto é máquina hoje em dia. [...] *Não vai ter jeito de você escapar da programação. Porque é ela que vai dar a função para as máquinas.*

Aluna2*: Oh... assim, pra mim, hoje se eu for falar assim: "- vai fazer muita falta na sua vida? Programação?" Eu acho que vai. Por exemplo: se eu quero trabalhar na área de projeto.* [...] *Como é que você vai discutir com o programador uma coisa que eu não sei?* [...] *Então, eu acho que é muito importante. [...]. Eu posso até não saber programar 100%, mas eu tenho que conhecer, porque se não, como é que eu vou discutir?[...] A partir do momento que você está num curso de Sistema de informação isso faz falta.*

Aluno4*: Eu classifico o bacharel em SI como um engenheiro civil de um prédio. O engenheiro, ele não vai colocar a mão no tijolo, fazer massa. Ele faz o projeto. Se ele não entender como fazer uma massa, por mais que ele não pegue, ele não vai garantir uma estrutura perfeita. Então, o bacharel em SI, tem que saber programar. Para poder no futuro, ou ele mesmo programar ou orientar alguém a fazer corretamente.*

Aluna5: *Olha eu acho que o curso [...], foi um curso direcionado a uma Engenharia de Software, mais voltado o projeto.Mas para projetar um sistema você tem que ter um mínimo de compreensão de programação. [...] Eu acho que ela* [programação] *é importante [...] Você precisa saber o mínimo para poder executar o projeto.*

A importância atribuída à programação pelos alunos e suas relações com a mesma, justifica, pelo menos em parte, seus conflitos estabelecidos com a área. Para eles, tornou-se inevitável a convivência com o fato de terem concluído um curso no qual o conhecimento de programação é indispensável, porém lhes escapa o domínio desse processo. Isso inviabiliza o exercício de suas atividades profissionais satisfatoriamente, implicando em insegurança e falta de realização.

5.2.2 – Os sentimentos de frustração e desânimo

Conforme o perfil da turma, o conflito experimentado pelos alunos com as UCs de programação os acompanhou durante toda sua trajetória acadêmica. Nas palavras dos educandos percebeu-se que mesmo com a troca de professores, mudança entre as linguagens de programação, persistiu a baixa aprendizagem e a certeza de que aquilo lhes seria prejudicial no futuro. A frustração esteve presente na fala de dois dos alunos participantes da pesquisa ao relatar sua trajetória nas UCs de programação:

Aluna2: *É... eu fiquei frustrada porque é ruim quando você se senta para... assim, por exemplo, você se senta no meio de uma porção de gente, que sabe programar, que conhece de programação, na verdade. Então a pessoa começa a falar... aí você ... sabe? A sensação que você tem é que você está boiando, ao mesmo tempo, você sabe do que a pessoa está falando. Mas você está ali boiando.*

Aluno1: *É, frustração. É bastante improdutivo em termos de grupo, de pessoas que juntam para fazer alguma coisa, ou mesmo você sozinho. A partir do momento que você fica frustrado toda vez,*

aquilo vai te atrapalhar, até o ponto de você não ter mais o interesse em fazer qualquer outra atividade.

Ao comentar sobre o curso de Sistemas de Informação, o Aluno4 manifestou sua afinidade e satisfação com o Curso, porém, destacou a existência de uma lacuna em seu aprendizado relacionada à área de programação:

Aluno4: *Olha o curso é maravilhoso. É tudo aquilo que eu gostaria que ele fosse. Pena que eu não aprendi o que eu gostaria de ter aprendido* [a programar um computador]*.*

O depoimento da Aluna5 reforça a idéia da baixa aprendizagem e aponta o início de um processo de desânimo relativo às UCs de programação:

Aluna5: *Eu gostaria muito de entender mais... Gostaria de mexer com ela* [programação]*, mas pelo desconhecimento e pela falta de compreensão de muitas partes você acabar ficando desanimada.*

Entendeu-se que a relação do grupo com as unidades curriculares de programação passou a ser referenciada de forma negativa, pouco produtiva e árida. O desgaste com a área resultou, ao longo do Curso, em um baixo nível motivacional.

5.2.3 – Os alunos e os fatores motivacionais ligados à aprendizagem de programação

A motivação de um aluno, em qualquer processo de aprendizagem, é fundamental para o seu sucesso. Guimarães e Boruchovitch (2004, p.143) afirmam que "um estudante motivado mostra-se ativamente envolvido no processo de aprendizagem, engajando-se e persistindo em tarefas desafiadoras, despendendo esforços, [...] buscando desenvolver novas habilidades de compreensão e de domínio". Os autores ampliam esse argumento elucidando que alunos motivados mostram-se entusiasmados durante a execução das tarefas propostas e que, ao concluí-las, revelam-se orgulhosos acerca dos seus desempenhos. No contexto deste trabalho, o referido sentimento encontra-se vinculado à atividade de programação de computadores e diz respeito ao *empowerment* (sentimento de eficácia) manifestado – ou não – diante dos desafios de conceber e estruturar com sucesso um programa.

Em um processo educacional é comum identificar alunos que variam entre dois pólos: aqueles que se aprofundam no conhecimento em busca de um embasamento que possa lhes valer no futuro – motivação intrínseca – e os que apenas se dedicam a cumprirem o "regulamento", ou seja, fazer provas e trabalhos visando à conclusão da

unidade curricular com um resultado minimamente satisfatório – motivação extrínseca (Guimarães *et al*, 2002).

Com o objetivo de apontar os determinantes motivacionais e os contextos promotores da motivação intrínseca, Guimarães e Boruchovitch (2004) valem-se dos trabalhos de Deci e Ryan (1985) e Ryan, Connel e Deci (1985), os quais fundamentam a Teoria da Autodeterminação. No contexto educacional, essa abordagem enfoca a promoção do interesse dos alunos pela aprendizagem e a segurança dos mesmos em suas capacidades, o que promove a valorização do processo educacional. Em linhas gerais, a Teoria da Autodeterminação envolve a satisfação das três necessidades inerentes à motivação intrínseca: a de autonomia, a de competência e a do estabelecimento de vínculos. A necessidade de autonomia está ligada à capacidade de agir com um mínimo possível de controle externo, ou seja: a ação auto-dirigida e ligada ao desejo pessoal de organizar a experiência e o comportamento. Procedendo desta maneira, o sujeito estaria envolvido em situações em que poderia experienciar a responsabilidade por seu desempenho, ou seja: se perceberia competente. O estabelecimento de vínculos com as atividades empreendidas seria o "pano de fundo" promotor da segurança inerente ao processo de desenvolvimento. Assim, em temos de motivação intrínseca, a autonomia, a competência e a vinculação afetiva seriam necessidades integradas e interdependentes.

Por outro lado, para um grupo que se limita à motivação extrínseca, o sentimento de apatia no processo educacional é praticamente inevitável, uma vez que seu compromisso está ligado estritamente a assegurar a "nota" para garantir sua aprovação e a conclusão de um curso. Neste sentido, nas entrevistas empreendidas com os discentes, buscou-se determinar o que levou cada sujeito a escolher o curso de Sistemas de Informação. Destacam-se os trechos abaixo:

> **Aluno1:** *[...] eu não queria fazer o curso [...] a partir do momento que eu fui e vi, que na minha turma tinha os meus amigos, que eu poderia dar certo me interessando mais pelo curso, eu decidi fazer o curso de Sistemas de Informação. [...]Eu não queria fazer nenhum curso de tecnologia. Eu queria fazer na área de farmácia mesmo, mas como não tinha aqui [...] eu fiquei para fazer o curso de tecnologia mesmo.*
>
> **Aluna2***: E quando eu fui tentar o vestibular, na verdade, foi uma coisa assim... Não foi planejada: O ano que vem eu vou tentar Sistemas de Informação. Não! Foi uma coisa meio no estalo.*
>
> **Aluna3:** *No começo eu não sabia nem o que era o curso. Ai, depois, eu comecei a pegar gosto pelo negócio [...] Não foi fácil. Muitas matérias eu tive dificuldade: programação e banco de dados [...]*

Fica evidente a falta de clareza e afinidade na escolha dos educandos pela área da computação. Entretanto, o Aluno4 e a Aluna5 manifestaram-se mais interessados pela área:

> **Aluno4:** *De início, eu entrei na faculdade por ter uma simpatia pela informática [...] Fui tentar o vestibular para fazer um teste como eu me sairia e consegui entrar na faculdade...*
>
> **Aluna5:** *Eu optei por fazer esse curso porque eu sempre gostei muito de tecnologia e informática [...] Depois, aconteceu de aparecer o curso lá* [na universidade Beta]*. [...] E aí eu vi a possibilidade de fazer o curso [...]*

Pode ser observado que nenhum dos cinco demonstrou um conhecimento mais aprofundado a respeito do que o Curso escolhido lhes exigiria. Os alunos não estavam certos do que viria ao longo de suas formações, ou ainda, não conheciam a estrutura e o perfil do egresso de um curso de Sistema de Informação.

Compreendeu-se, portanto, que nos casos relatados, o prazer de manipular a tecnologia foi dificultado, uma vez que os sujeitos envolvidos não estavam em um processo que lhes era plenamente afim. Isto foi entendido como uma das prováveis origens do problema com a programação. O simples "gostar da área" precisaria evoluir para uma dimensão mais ampla: a da dedicação e pesquisa, a do prazer em usar a tecnologia e, finalmente, a do aprender a programar um computador. Em outras palavras: uma escolha mais consciente e intrinsecamente motivada levaria os sujeitos a terem maiores condições de alcançar melhores níveis de aprendizado, desempenho e interesse pela área. Nestes termos, os alunos teriam assumido uma atitude "mateticamente" amadurecida, o que teria os conduzido para uma melhor apropriação de conceitos e da prática de programação.

Não se pode omitir que o processo educacional é norteado, predominantemente, pelo método transmissivo de conhecimento e que este modelo acompanhou os sujeitos desde os primeiros momentos em que foram alfabetizados. Dessa forma, ao longo de suas formações, os alunos não foram instigados a expressar suas idéias de forma lógica, ao trabalho cooperativo e à resolução de situações problema. Nos moldes tradicionais, não existe o estímulo à pesquisa, ao debate e ao posicionamento crítico. Os conteúdos trabalhados são retalhados, ou ainda, tratados

desconexamente. Não se valoriza o tempo como elemento maturador da aprendizagem e do aprimoramento de idéias. Percebeu-se, aliás, nesses fatores parte da justificativa do choque sofrido pelos educandos envolvidos com a programação de computadores, que a tudo isto requer. Ficou, portanto, a cargo dos alunos (re)agirem e (re)direcionarem a sua forma de estudar, assumindo uma nova postura frente à aprendizagem. Isto foi constatado quando os mesmos foram questionados a respeito da postura desejada ao aluno de programação:

> **Aluno1:** *O aluno ele tem que ter bastante interesse. Porque não é fácil mesmo não. Não é uma matéria que você vai ler e vai entender logo, logo. Não é igual a você ler um texto. Você tem que ler e procurar a assimilar o que vai fazer e juntar com lógica, com matemática e outras coisas mais. [...] Para o aluno, ele tem que estar bastante preparado e ter bastante interesse na matéria mesmo.*
>
> **Aluna3:** *Você tem que correr atrás, não pode ficar esperando só de quem está ensinando. Acho que faltou, até de mim mesmo, pelo fato de ter dado uma esfriada, uma desanimada, deixou um pouco de lado aquele negócio de todo dia sentar, entendeu?*
>
> **Aluna2:** *[...] acho que você também tem que fazer a sua parte. Não adianta. Achar que o professor vai te ensinar tudo. Não adianta.*
>
> **Aluna5:** *O aluno tem que ter no mínimo interesse em aprender. [...] se o aluno não quer aprender ele não vai aprender mesmo. Ele* [aluno] *tem que correr atrás e procurar estudar também.*

Como se observa, indicar a necessidade de avanço no trabalho pedagógico dos professores, em momento algum aponta para o descompromisso e a não necessidade de empenho permanente dos alunos. Nesse sentido, o aluno de programação pode ter a oportunidade de experimentar uma sensação de *empowerment*, ou seja, deliciar-se com o prazer e com o orgulho de ter conseguido construir algo ligado ao seu interesse, a algum campo de aplicação e que compreende desde o planejamento inicial até a implementação final. O *empowerment* prevê o engajamento discente (*head-in*), o que se traduz no interesse pela prática de programação (*hands-on*). Entretanto, se um sujeito se envolve em um meio que não lhe gera satisfação e interesse e que não o instrumentaliza para a ação-reflexão, ele não tem ali nem os instrumentos e nem a motivação para se realizar.

Nessa dinâmica de (re)adequação do papel do aluno para o sucesso com a programação, a dimensão motivadora se faz indispensável, no sentido de equilibrar o esforço empreendido, os resultados futuros, a efetivação da aprendizagem e a construção

do conhecimento. Onde, na concepção do aluno, estaria(m) o(s) elemento(s) motivador(es) desse processo de ensino-aprendizagem?

> **Aluna3:** *Eu acho que primeiro o professor teria de motivar mais o aluno. Procurar ver uma forma mais fácil, mais simples de atender cada pessoa. Porque cada um tem um grau de dificuldade diferente. E o aluno também correr atrás.*
>
> **Aluna5:** *Eu acho que para motivar o fundamental seria ter um professor que tivesse uma boa didática. Entendeu? Desde o primário, eu nunca gostei de matemática, aí quando eu mudei de colégio, eu tive uma professora que explicava e eu fui entendendo tudo. Eu acabei sendo uma das melhores alunas de matemática no final do ginásio. Era a forma como ela explicava. [...] Não é que eu não gostasse de matemática, quem passava a matéria, talvez não soubesse transmitir direito.*
>
> **Aluna2***: Se o professor conseguir entender, que ele precisa mostrar pro aluno primeiro o que vai acontecer, pra depois ele ensinar o caminho que o aluno vai ter que percorrer pra chegar até lá, é uma forma de motivar o aluno. E aí o quê que acontece? Com isso, o grupo todo vai ficar motivado, porque você quer fazer, quer mostrar para o outro: "Olha eu consegui!", mas como é que você fez? E você ensina pro outro... é mais uma forma de você aprender e de você ficar mais feliz. "- Nossa eu consegui e eu ainda consigo passar!". Não adianta você aprender e depois não conseguir explicar nem o que você está fazendo.*

Os alunos colocam o professor num lugar especial no processo de ensino-aprendizagem de programação: além de ser o responsável por estruturar e criar as condições para a apresentação dos conteúdos programáticos, os alunos esperavam que a forma de conduzir o processo de aprendizagem extrapolasse a instrução direta, sendo necessária a incorporação de uma dimensão didática no seu fazer. Este indicativo encontra respaldo no trabalho de Guimarães e Boruchovitch (2004), o qual preconiza que a motivação intrínseca do aluno não é o resultado direto da instrução, mas que essa pode ser influenciada pelas ações dos professores:

> [...] alunos de professores com estilo motivacional promotor de autonomia demonstram maior percepção de competência acadêmica, maior compreensão conceitual, melhor desempenho, perseveram na escola, aumentam sua criatividade para as atividades escolares, buscam desafios, são emocionalmente mais positivos, menos ansiosos, buscam o domínio e são mais intrinsecamente motivados [...]. (GUIMARÃES e BORUCHOVITCH, *op. cit.*, p.148.)

A ação docente apresenta-se, portanto, como um importante elemento motivacional inerente ao processo de ensino-aprendizagem de programação de computadores.

Também os Alunos 1 e 4 destacaram o professor como elemento motivador:

> **Aluno1**: *Às vezes o fato de um professor ter uma didática interessante, às vezes poderia ser uma motivação. Ele próprio ser um motivado para explicar. E a gente, ia se tornar com isso. Porque um professor infeliz, ele tem alunos infelizes. Agora, um professor que é alegre e dá aula com felicidade, todo mundo ia chegar e falar: "A aula daquele cara é super produtiva. Ele conversa, ele explica e eu entendo. Tudo que ele faz é interessante para aula. O que ele traz ele estudou, e a gente sabe que ele está dando. E a gente vai em casa estudar para acrescentar na aula do professor". Isso ia ser uma motivação interessante.*
>
> **Aluno4:** *É... Por mais que eu buscasse idéias, condições junto com os colegas, a faculdade nunca dava condições para a gente. Porque profissionais bons, professores da área de sistemas de informação existem por aí. Só que a maioria deles já tem o seu vínculo no mercado. E uma vez que precisando buscar profissionais, aqueles que foram encontrados, eles não tinham uma didática, assim... Não tinham uma experiência boa [...] eu não estou culpando os professores somente. Eu acho que faltou um pouco mais de ajuda da faculdade. Se tinha uma deficiência vamos bater em cima dela.*

Os discursos dos alunos mostram o professor e sua forma de atuação como fatores motivacionais para a aprendizagem de programação. O Aluno4 acrescenta ainda a necessidade de uma estratégia administrativa na seleção dos professores. Apesar da consciência de que deveriam fazer a sua parte, suas falas indicam que depositavam na figura do professor as esperanças de apoio na construção/efetivação do conhecimento.

5.2.4 – À espera de um professor construcionista

Diante da ênfase posta na figura do professor, buscou-se conhecer as características desejáveis para o profissional que trabalha com a docência de programação. Afinal, qual seria o perfil desse profissional?

> **Aluno1:** *Uma pessoa que seja capaz de estudar e entender por completo aquilo que ele está querendo passar. Não que ele tenha que entender tudo, mas pelo menos aquela matéria que ele está dando no dia, procurar saber todas as saídas para uma explicação, para uma pergunta que um aluno fizer, ou uma dúvida... Ele dever saber exemplificar e fazer a pessoa entender naquele momento. Não deixar a pessoa pensar e depois ver o que vai acontecer com ela.* [...] *Eu acho que um professor [...] tem que ser uma pessoa, que assimila as dificuldades de todo mundo. Porque a partir do momento que tem algumas pessoas tem a facilidade com a programação, outros tem as dificuldades delas. Mas nem por isso elas estão desinteressadas. Elas ficam desinteressadas a partir do momento que aquilo vai acumulando e chega ao ponto dela já perder o fio da meada.*
>
> ***Aluno4:*** *Um bom professor na área de programação, a primeira coisa, é conhecer as ferramentas. Ele tem que ter domínio da sua ferramenta. [...] [o professor] pega uma ferramenta, estudou sobre ela, ouviu dizer dela, quis aprendê-la primeiro, para depois jogar para os alunos. Os alunos vão trabalhar com ela. [...] Os professores [...] tinham um conhecimento técnico da ferramenta, mas eles não tiveram um conhecimento didático para passar para os alunos essa técnica deles.*

Os Alunos 1 e 4 destacam a necessidade do domínio dos assuntos tratados, bem como a capacidade de tornar os conteúdos inteligíveis a uma turma heterogênea, utilizando exemplos e atentando para as dificuldades que se estabelecem. Para eles, esse professor deveria ter afinidade com a área não apenas no sentido técnico, mas também no pedagógico. Em outras palavras, para dar aula de programação o professor precisaria dominar o conteúdo teórico, conhecer as linguagens e ambientes de programação (ferramentas) e, não menos, ter gosto e formação para o ofício docente. Pedagogicamente, a gerência das diferenças na aprendizagem, também é fator relevante. A Aluna2 e o Aluno4 ampliam a discussão, introduzindo novos elementos:

> **Aluna2:** *[...] a primeira coisa é ter conhecimento. Tem realmente que ter embasamento daquilo que ele está falando. Não adianta falar para os cocos. Ele tem que estar falando aquilo que é a verdade. Segundo, acho que tem que ser um professor que tenha a vivência [...] Não adianta ele ensinar uma coisa que ele, realmente, não sabe fazer na prática. Porque é muito fácil... você falar...eu pego, dou uma lida numa revista, alguma coisa, comento na sala de aula ... faz aqui pra mim... não dá para fazer nesse sentido.* [terceiro] *aquele professor que tenta motivar o aluno, mostrar para o aluno que aquilo que vai ser ensinado para ele é legal, que vale a pena aprender, que produz alguma coisa, porque eu acho que não tem nenhum conhecimento, em área nenhuma, que não produz alguma coisa. Que dê um resultado.*
>
> **Aluna3:** *Eu acho que o professor tem que estar mais preparado para estar ensinando ao aluno.* [ou seja, que prepare uma] *aula que incentive, que motive, que chame mais atenção do aluno, que seja uma aula mais clara, mais fácil de entender. Que ele procure mostrar de uma forma mais simplificada.*

O aspecto "vivência prática de programação" é um elemento desejável ao professor. Compreendeu-se que esse atributo traria, para a sala de aula, a possibilidade de análise e contextualização do conteúdo por intermédio de situações práticas, advindas de fora do ambiente acadêmico. Mais uma vez, percebeu-se que os fatores motivacionais dos alunos estão associados à atuação do professor na condução das UCs, que deveria ser clara e acessível a todos.

Uma vez traçado o perfil para o professor de programação, foi levantado junto aos alunos qual seria o seu modo de ação. Como esse professor deveria atuar a fim de favorecer a aprendizagem de seu grupo de alunos?

> **Aluna3:** *Entrou um professor novo que achava que a turma estava no mesmo nível que ele, então, tacava um negócio que ninguém entendia, dava a aula dele e acabou. [...] Até, a gente ficava bobo de ver, a turma inteira, o "Fulano" chegava... Ele tinha uma cabeça... Não pegava uma folha... passava tudo no quadro... a gente achava aquilo ali fantástico. Né? Mas com relação a ele poder estar passando a aula ali para a gente... Não tinha como, porque ninguém conseguia entender*

nada que o homem estava falando. Alguma coisa a gente conseguia entender, porque ele falava mais claro.

Aluna5: *o próximo que entrou* [professor novo] *era um cara que eu acho que entende muito, uma cabeça muito rápida de raciocínio... e ele sabia muito, mas eu acho que ele não tinha didática nenhuma. [...] sabia muito para ele, porque ele chegava despachava no quadro aquele raciocínio todo, aquela programação toda e ia passando, ia passando e ele não queria saber se você estava pegando ou não. Ele queria cumprir o planejado, mas independente do pessoal estar pegando ou não.*

Nas declarações acima, configura-se a "aula espetáculo", no sentido de uma demonstração de domínio técnico do conteúdo do ementário. Ou seja: o professor funcionava ao molde de um equipamento pré-programado, como um computador, que vai a uma unidade de armazenamento de dados, busca um conteúdo X, segue um roteiro estrito para cumprir sua demonstração, não introduz novos elementos ao contexto e dá por terminado o processo. Esse é um caminho, que na visão daqueles alunos, não deveria ser seguido em um ambiente de aprendizagem, uma vez que eles expressam sua admiração pela capacidade de reproduzir informação, mas contestam a forma de apresentá-las aos outros. A Aluna3 foi veemente ao afirmar que ninguém entedia o que se tentava passar. Percebeu-se que não houve preocupação com a forma de se processar o conteúdo programático e também em como adequá-lo ao nível dos alunos.

O Aluno1 acrescenta outros questionamentos sobre o modo de atuação dos professores:

Aluno1: *Eu não sei o que acontece, mas quando ele teve esse contato, o próprio professor, com a matéria, eu acho que ele teve no mesmo método que ele estava tentando passar. Ele não teve esse método mais facilitado. Ou então, um entendimento com mais facilidade para a programação. Foi o método que ele aprendeu e eu acho que era o método que ele achou que ia ser bom para passar para os seus outros alunos. E... O que às vezes atrapalha. Porque ele próprio se priva do que ele poderia criar para estruturar, para uma outra pessoa entender. [...] A partir de um momento que ele aprendeu de um mesmo jeito, ele vai querer passar do mesmo jeito que ele aprendeu.* [...] *Existem vários métodos, existe o fácil e existe o outro que é muito complicado [...]*

Pesquisador: ***E por que você acha que eles fazem isso?***

Aluno1: *[...] eu acho que pelo fato de não ter tempo para estruturar uma aula e eles já terem essas aulas programadas de alguma outra pessoa que passou e ele pegou. [...] Porque dá para você sentir quando o professor pegou aquela aula, aquela matéria que ele está dando, estudou na casa dele e falou assim: "Não, o jeito que eu vou dar essa aula, para essa turma, vai ser desse jeito. Não é o mesmo jeito que eu dou para uma outra turma que já sabe muito". Ai você sabe quando a pessoa falou assim: "Esse professor pensou e estruturou essa aula para a minha turma. Eu tenho certeza disso".*

De acordo com Aluno1, a ação docente requer planejamento, criatividade, envolvimento pessoal, adequação do conteúdo ao perfil da turma com a qual ele será trabalhado. O Aluno1 reprovou o modelo de aproveitamento de "aulas padrão", supostamente cabíveis a qualquer turma, denunciando o caráter repetitivo e impessoal das estratégias docentes adotadas em salas de aula.

A Aluna5 colocou a atividade docente como um dom ou ainda como um ofício que é intrínseco ao sujeito-professor:

> **Aluna5:** *Na verdade, o que foi passado a meu ver, foi muita teoria... e pegava a teoria e despachava ela naquela programação. Não teve assim... um exemplo, uma analogia para que a gente entendesse mais a lógica da programação...*
>
> ***Pesquisador? E por que você acha que o professor fazia assim?***
>
> **Aluna5:** *Primeiro falta de tempo dele. Eu acho que a vida dele era tão corrida que ele não tinha tempo de parar e pensar uma forma de fazer com que os alunos entendessem aquilo. [...] Uma coisa que a gente sente: isso é da pessoa. Não adianta querer impor. O prazer de fazer o outro aprender está na pessoa.*

Já o discurso da Aluna2 deixa evidente a necessidade do rompimento com a cultura de rígidos (*hards*), fazendo coro com a contestação apresentada por Papert.

> **Aluna2:** *Eu acho que para a programação o principal é despertar no aluno que... Não é que ele tem que decorar comandos, linhas e palavrinhas do inglês. [...] o professor não tem que enfiar isso na cabeça do aluno. O professor tem que mostrar para ele é que a lógica é que funciona. Porque quando você tem a lógica do negócio, os comandos vem assim: " – Mas como é que eu faço pra fazer isso repetir? – Eu uso esse comando..." Ai, encaixa as linhas, os comandos, tudo encaixa ali no raciocino do aluno. Só que... no começo, ensinaram primeiro, para gente, os comandos e depois ia mostrar onde os comandos iam entrar. E não é... Tem que mostrar como é que funciona. [...] Mostrar o raciocínio do negócio mesmo, pra depois enfiar os comandos no raciocínio do aluno. [...] Porque quando a gente chega é Introdução à Lógica...começa a ensinar V com F, F com V... pra mim não funcionou não... não teve Introdução à Lógica nenhuma. Pra mim eu fui tomando pavor de Lógica. Sabe?*

O tecnicismo predominante na Ciência da Computação, área de conhecimento na qual se enquadra a programação de computadores, parece exercer uma influência muito forte no comportamento rígido dos professores da área. Entendeu-se, em conformidade com o pensamento de Papert, que seria necessário o desenvolvimento de uma cultura mais leve (*softer*) para se tentar promover o processo educacional. Ou seja: a rigidez imposta pelo formalismo matemático da computação precisaria ser adequada a uma forma menos abstrata, mais inteligível e atraente para o aluno. Nesse sentido, destaca-se um trecho onde a Aluna3 confirma essa necessidade:

> **Aluna3:** *Teve uma pessoa que me ajudou muito, que foi o "X"* [aluno companheiro de sala]*, [...] muita coisa que eu aprendi foi porque ele explicava assim... do meu jeito, simples. Então, era uma linguagem que eu entendia. Muita coisa eu entendi pela ajuda dele. Fora ele... mais ninguém.*

O aluno declarou que ele tinha um nível de compreensão dos conteúdos que precisaria ter sido diagnosticado pelo professor. Entretanto, essa demanda foi suprida por um companheiro de sala, que fez valer o conceito de ZDP e atuou de forma a contribuir com a aprendizagem daquele sujeito.

5.2.5 – LOGO: uma alternativa para o ensino de programação

Nesta seção, destacam-se as impressões iniciais dos alunos sobre o LOGO e suas percepções em relação à Linguagem: seus pontos fortes e fracos, sua utilidade no processo de aprendizagem e as mudanças causadas pelo contato com o *software* durante a unidade curricular "Tópicos Especiais".

LOGO: as primeiras impressões são as que ficam?

Mediante o perfil traçado da turma na sessão 5.1, a oferta do referido curso estava diante de um grande desafio: trabalhar com alunos relutantes frente à programação e que, entre muitas outras matérias, estavam concluindo sua graduação. Era de se esperar certa apatia, descrença em relação ao que estava por vir nas 40 horas do curso ministrado. Durante a entrevista com os participantes, percebeu-se que algumas impressões iniciais convergiam com essa expectativa. Destaca-se que nenhum dos entrevistados havia tido um contato prévio com a linguagem LOGO, desconhecendo totalmente suas potencialidades.

> **Aluna2:** *De primeiro momento você olha e fala assim: "Nó... não acredito... nessa altura do campeonato ficar desenhando casinha, puts... com tanta coisa pra eu estudar lá em casa, eu vou ficar vendo essa tartaruga ficar rodando pra lá e pra cá?". De primeiro momento eu tive essa impressão. [...] Mas depois que você começa a ver, a entender o negócio...*

> **Aluna3:** *No início eu achei bobinho. Eu falei: "- Ah não!". Mas depois: "- Bobinho nada! Tem coisa aqui que eu não sei fazer não".*

As primeiras impressões dos alunos conduziam à idéia de perda de tempo, ou ainda, que aquela Linguagem nada acrescentaria ao que aqueles alunos concebiam em termos de programação. Entretanto, percebeu-se que essas concepções eram transitórias e assumiriam outra perspectiva. O fato é reforçado nas falas de dois outros alunos:

> **Aluna5:** *No principio, assim, a primeira e a segunda aula eu achei que fosse para criança, mas depois que eu vi que eu podia criar com ele... e criar coisas que crianças não criariam... Então seria uma linguagem fácil, para aprender programação, que um adulto maduro, conseguiria enxergar o que era realmente a programação.*

> **Aluno4:** *Tão logo você apresentou o programa para a gente, ora nenhuma eu olhei com crítica negativa. Nunca interpretei que aquilo lá era brincadeira de criança. Sempre interpretei que era brincar de forma [...] séria. Então, o SuperLogo, ele foi uma programação para criança enquanto aprendizagem* [aprendiz]*: então, nós éramos crianças naquele momento.*

O choque proporcionado pela cultura *softer* idealizada por Papert fica evidente no posicionamento dos alunos quando tomaram contato com o LOGO. Formandos e acostumados a ferramentas concebidas para um universo completamente diferente do que lhes surgia, seus sentimentos foram balizados por uma perspectiva ligada ao universo infantil: uma tartaruga robótica que cumpria ordens não poderia estar associada a uma ferramenta de programação, outrora impenetrável, complexa e definitivamente fora de seus domínios.

A declaração apresentada abaixo despertou a atenção, pois faz referência ao trabalho realizado por Papert ao idealizar o LOGO. O aluno dedicou suas observações ao contexto de criação da ferramenta, ficando os aspectos da linguagem de programação em segundo plano.

> **Aluno1:** *De primeira, [...], eu me interessei pela história que você introduziu do [...] Seymour Papert, né? Eu achei interessante o que ele pensou mesmo de dar um computador para cada criança, que – às vezes – não é questão de robotizar uma juventude, mas em si, introduzir o que virá pelo futuro. Porque a criança tendo o contato com a tecnologia, ou então, até mesmo com a programação, que é facilitada pelo LOGO,[...] às vezes, a criança ia adquirir aquele conhecimento com muita mais facilidade do que se ele fosse colocado, essa estrutura na mente dele, a partir dos quinze anos, ou uma coisa assim. Porque a criança, quando ela está crescendo, todo o entendimento que ela aprende de criança é que ela vai levar para o resto da vida. [...]. É o que eu achei que foi interessante no início. [...] A tal ponto que ele colocou todo esse pensamento dele, em uma ferramenta simples, com uma tartaruga que, às vezes, uma pessoa enxerga como lenta, [...]. Não sei se foi o que ele pensou, mas foi o que eu achei. Que ia ser fácil de uma pessoa pegar e não fazer repúdio daquilo, mas sim falar: " – Puxa achei legal... vamos ver o que isso ai faz... deixa eu fazer uma reta para eu ver o que isso vai me acrescentar." Ou " – Ah, isso é uma bobeira essa tartaruga andando". Mas no fim, a pessoa começa a ver que aquilo lá tem uma lógica, por trás daquilo, vai te ensinar com facilidade.*

Com exceção da Aluna5, percebeu-se que no final dos discursos dos alunos as expectativas apontam para uma dimensão que supera as primeiras impressões. Perda de tempo ou tempo a ser recuperado? Aquelas impressões mudariam e os alunos começariam a ver ali uma oportunidade para rever ou mesmo aprenderem programação. A Tartaruga passaria a ser a companheira da aprendizagem e a viabilizadora do acesso à programação de computadores.

LOGO: é possível aprender a programar!

A seção anterior destacou que os primeiros momentos de contato com a linguagem LOGO foram acompanhados de uma postura de desconfiança quanto à utilidade da ferramenta. Entretanto, com o desenvolver do curso, com a apropriação do vocabulário sintático e, principalmente, da semântica LOGO, os alunos começaram a vislumbrar uma nova realidade em programação. O estudo de perfil realizado na seção 5.1 indicou que a maioria dos alunos da turma não aproveitou satisfatoriamente os conteúdos relacionados à programação de computadores ao longo de sua trajetória acadêmica. Naquele cenário, seria possível ter o entendimento e conseguir construir programas de computador?

> **Aluno1:** *É...* [LOGO] *faz uma analogia assim de fácil, porém complexo. Você pode fazer uma coisa simples, como você pode fazer uma complexa. Ele te dá o espaço para você ir g r a d a t i v o. Se você quiser entender aquilo sozinho, você entende. Você vai fazendo até o ponto de você ir criando cada vez mais e chegar ao ponto de você falar: - Poxa, agora eu faço isso aqui, a mesma coisa que eu fazia numa linguagem que não tem o grafismo ao mesmo tempo, eu consigo fazer nessa linguagem, só que com outro tipo de estruturação da linguagem mesmo. É... no caso do LOGO, por ele introduzir mais fácil, como: "Para frente e tal". Uma coisa de entendimento mais lógico...*

Para o Aluno1, LOGO funcionaria como uma ponte entre o abstrato e o concreto, proporcionando o sentimento do "fazer compreendendo". O *Empowerment* também passou a ser acessível mediante a programação com essa Linguagem.

A Aluna2 fez uma análise do comportamento de um companheiro de sala, com o qual se relacionava e trabalhava em grupo:

> **Aluna2:** *Eu falo como companheira agora: para o "*fulano*" foi uma experiência fantástica. Porque pior do que eu, pra ele... Ele falava: "Oh, eu não sei uma linha. Se você me perguntar direito o que é variável eu nem sei mais. Eu não lembro. No sétimo período". De repente veio esse trem aí* [o

> LOGO]*, o* fulano *vibrava, acho que mais que todo mundo. Bem mais do que eu, porque assim, ele ficou devendo JAVA, passou um monte de problema. Enfim, pra ele foi mais desmotivador* [as UC de programação] *do que para mim. Mas, em contrapartida para ele, no final, deu assim: uma esperança. De que pode ser diferente, se a pessoa estudar e começar do que é mais fácil, do que – na minha opinião – você pode visualizar antes, que funciona. Só que infelizmente veio, assim, no último período, com pouco tempo, pra gente poder estar despertando mesmo essa coisa de você fazer.*

Nas palavras da Aluna2, LOGO foi a oportunidade que seu amigo de sala teve, já no final de sua graduação, de retomar a aprendizagem de programação. Segundo ela, aquele sujeito demonstrava não dominar os conceitos básicos de programação. Entretanto, em contato com a nova Linguagem, ele passou a contar com uma nova chance na programação. A Aluna2 destacou ainda, que apesar da validade do LOGO e do curso elaborado, o esforço empreendido foi muito tardio (último período), merecendo mais tempo para obtenção de uma apropriação ainda maior.

O Aluno4 e a Aluna5 manifestaram o quanto estavam felizes por poderem "aquecer", no sentido proposto por Papert, seus conhecimentos de programação:

> **Aluno4:** *Como uma criança não sabe ler, nós éramos crianças que não sabíamos programar. [...] Para mim foi perfeito, [...] Aquela teoria inicial, que todo aluno precisa, ela dá suporte. Eu vou arriscar a falar 100%.*

> **Aluna5:** *Eu até achei que fosse um programa de editoração gráfica [...] e como eu adoro editoração gráfica, ascendeu a luzinha de interesse! Só que com o decorrer do tempo eu descobri que através dos desenhos eu estava entendendo o que era programação***.**

Notou-se o surgimento de uma nova possibilidade com a programação para alunos adultos. Naquele momento, o LOGO não era mais uma ferramenta pueril. Uma nova fase começava para o grupo: programação era uma algo acessível! Eles poderiam se apropriar, compreender e, finalmente, construir programas de computador. Percebeu-se o início de um rompimento com o repúdio à área de programação. Essa constatação levou ao questionamento seguinte: em termos motivacionais, o que o trabalho com o LOGO proporcionou à trajetória acadêmica dos alunos?

> **Aluno1:** *É... Faz com que você tenha motivação para criar. Ou então, se te falar: "Faz um exercício que faz criar um círculo", [...] Eu não ficava só no círculo, por quê? Você sabe que aquele círculo ali já foi fácil para você, eu já passei aquilo ali, para mim já era... O círculo para mim eu já fiz, agora eu quero fazer 1000 círculos e um dentro do outro e fazer um raio saindo do meio. Você mesmo já cria aquela expectativa, já cria aquela idéia na sua mente até o ponto de você conseguir executar. A partir do momento que você executa aquilo: "Ah isso aí eu já fiz, agora eu quero já é outro lugar, outra coisa".*

O aluno associou o LOGO a uma linguagem que proporciona meios para a expressão criativa de idéias, isso por intermédio dos seus recursos gráficos que funcionariam como um elemento capaz de despertar a motivação em programar nos estudantes. Junto à dimensão criativa, a programação com o LOGO propiciaria o *empowerment* e o querer ir além.

A Aluna2 também destacou a aprendizagem visual proporcionada pelo LOGO e introduziu a dimensão reflexiva sobre idéias, inerente à atividade cognitiva de programação:

> **Aluna2:** *Não, igual o dia que você mostrou uma florzinha... "nó, mas que bacana, mas como é que ela* [a Tartaruga] *faz aquilo?". Então, você já começa antes de você sentar na frente do computador, a raciocinar como que você vai fazer para ela fazer assim, assim, virar e descer. Antes de ir pra máquina você já está com o raciocínio na cabeça. Então, assim, faz com que você pense e aí, desperta a lógica do trem. Porque se a tartaruga fez aquele desenho, que ficou assim, parecendo uma flor, ela teve que virar para um lado, virar para o outro um monte de vezes, quantas vezes ela vai ter que virar? Você já começa a pensar. Às vezes as idéias estão meio perdidas assim, entendeu? Mas na hora que você senta lá pra fazer, mesmo no papel, você vai colocando as idéias, você olha e fala assim: "Não, mais se for para virar para cá, então isso aqui é para direita... Não! Então, eu fiz errado aqui. Oh, tem que virar tantas vezes para a direita". Nesse sentido, te desperta porque você viu ali. "Ah, ela foi pra frente", mas como é que ela vai pra frente? Você vê que, às vezes, você falava assim: "como é que faz para apagar?", "– Ah! Use borracha", era uma coisa assim: despertava o entendimento mesmo. Sem precisar você ensinar os comandos eles vinham. Sabe? Despertava aquele... tipo como se fosse um instinto mesmo, como se a gente já tivesse com o sangue da programação.*

Para a Aluna2, LOGO criou condições de refletir sobre o "como?" e o "por quê?" de um programa de computador. LOGO foi um facilitador da estruturação lógica de idéias, fortemente destacadas no seu aspecto gráfico. Além disso, funcionou como um facilitador do aprendizado, uma vez que sua sintaxe e semântica são simplificadas.

Os três últimos alunos assumiram uma postura de mudança de visão a respeito da programação. O que antes era inacessível, ou ainda, muito abstrato, com o LOGO assumiu uma nova dimensão: a do entendimento e compreensão.

> **Aluna3:** *Eu vi que não era o bicho de sete cabeças igual a gente estava pensando. Há chances de estar conseguindo fazer.*

> **Aluno4:** *Primeiro me fez aperfeiçoar no conceito do que significa lógica. [...] Quando eu peguei o SuperLogo eu passei a observar: "ah, então aquilo ali que é um parâmetro? É assim que é um parâmetro? Nossa! [...]". Ele fez com que eu entendesse melhor programação.*

> **Aluna5***: Eu vi a possibilidade de eu aprender programação, que eu achei que era impossível, [...] que seria muito complicado eu pegar programação mesmo. E eu vi que não era tão difícil assim. Basta a forma de que é apresentada para você.*

No geral, as falas dos entrevistados indicam que o proveito tirado do LOGO foi considerável. Suas perspectivas se ampliaram com o conhecimento da Linguagem, o que retomou o processo motivacional para a construção de programas de computador. De conceitos simples, passaram a se interessar por situações que envolviam maior nível de complexidade, o que é tido como desejável em um processo educacional. O LOGO e o ambiente de trabalho no qual o curso foi ministrado representou para os alunos a retomada da possibilidade de se programar um computador.

"LOGOEmpowerment": as mudanças no comportamento dos alunos frente à programação de computadores

O resgate da oportunidade de experimentar a programação em uma visão diferente da até então vivenciada, proporcionou para aqueles sujeitos uma nova forma de se permitirem um contato com a tecnologia e, essencialmente, com uma linguagem de programação.

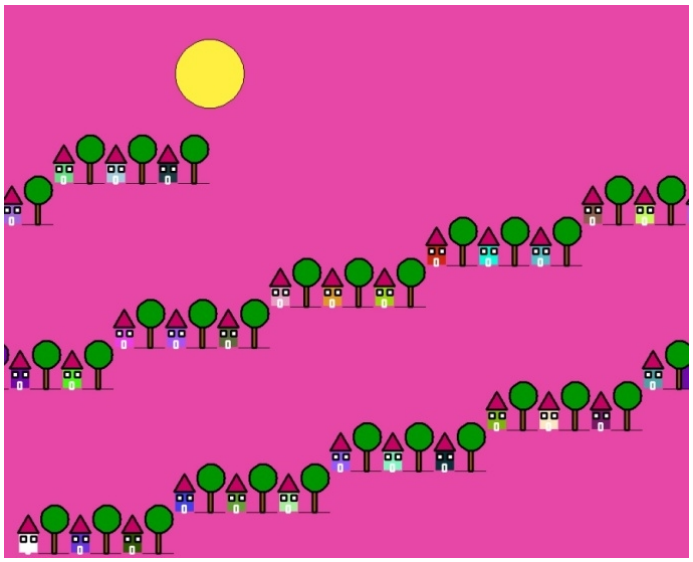

Figura 5.7 – Um bairro em LOGO programado pelo Aluno1.

A figura 5.7 mostra o resultado gráfico de um trabalho do Aluno1. Por trás da aparente simplicidade do desenho, configuram-se importantes conceitos de programação de computadores, tais como estruturas de fluxo de controle de dados, parametrização de procedimentos, uso de randomização (sorteio) para uso de cores e chamadas de procedimentos dentro de outros procedimentos. Chama-se a atenção para o fato de que essa gama de conceitos estava "resfriada" na idéias do aluno e o LOGO

conseguiu “esquentá-la” novamente. Foi selecionada essa produção dado o efeito de *empowerment* experimentado pelo aluno, que acabou por contagiar muitos outros. Finalmente era possível programar um computador! As palavras do Aluno1 enfocam o orgulho de uma possível produção de *software*:

> **Aluno1:** *[...] eu queria fazer um software. Eu e o* “Beltrano” *lá da minha sala. É... um jogo para um celular. Não ia fazer um jogo com todas as suas qualidades, ou para rodar e vender. Mas era para fins de si próprio, para falar: “– Eu fiz um jogo para o meu celular”. Ou então: “– O jogo que está rodando no meu celular foi eu quem fiz. Eu fiz esse jogo aqui a partir do momento que eu consegui entender uma programação e eu vi que eu era capaz de entender essa programação complexa para executar no meu celular e compartilhar com as outras pessoas”. Porque você vê o seu software: “– Ah, aquela pessoa joga o meu jogo”. Isso ia ser interessante. Então,* [o contato com o LOGO] *me motivou a querer a linguagem J2ML que é para celular, né? E fazer essa coisa...*

Procurou-se diagnosticar entre os entrevistados, como eles situavam a utilidade da linguagem LOGO no processo de ensino-aprendizagem de programação. Neste sentido, destacam-se os seguintes trechos:

> **Aluna2:** *Ela é útil. Muito útil.* [taxativa]. *Mas eu acho que ela seria muito mais útil, se ela fosse apresentada no começo. Para desmistificar uma coisa: que programar... “nossa se o cara é programador, o cara veio de outro planeta, veio de outro mundo, o cara cria uns negócios ali...um botãozinho, o cara é de outro planeta”. Eu acho que essa é a sensação. Então, quando você vai desmistificando aquilo, que não é assim, que é uma matéria como outra qualquer, que você aprende, que você executa e que funciona. Igual na matemática: você aprende, executa e ela te dá um resultado, sabe? Aí fica mais fácil. [...] Você faz um pedacinho, executa lá, funcionou... E você vai aumentando ali o que você quer fazer, mas podendo ver passo a passo, você tem uma resposta dele quando você erra, que te faz pensar o que você fez de errado.*

Para a Aluna2, LOGO se consolidou como uma linguagem acessível e adequada à introdução à programação. O rompimento com a cultura de rígidos, que exclui os sujeitos da possibilidade de aprenderem a programar, cedeu espaço a uma nova forma de se apropriar dos conceitos de programação: a forma *softer* de construção de programas idealizada por Papert. Para aquele aluno, não era mais preciso ser possuidor de um dom especial para se ter a compreensão de programação.

O Aluno1 e a Aluna5 tiveram impressões similares à validade do LOGO enquanto ferramenta do aprendizado de programação:

> **Aluno1***: Ponto forte do LOGO é você conseguir fazer a escrita da programação e já ter o resultado no grafismo. Que, às vezes, é um resultado mais óbvio em cima do que você está querendo estruturar. Mesmo que seja uma advertência, que você não fez uma coisa certa, mas pelo menos ele já te deu o resultado que você queria. “– Ah, não é isso!”. Então eu já sei onde eu vou mudar. Ai você fez: “– Ah, é isso!”. Ai ele te apresenta o êxito na tela: “– Ah, eu consegui. Era isso que eu*

queria que ele fizesse!". É uma coisa simples, mas que no fundo vai é te dar, ou acrescentar é... Uma facilidade de entendimento no que você se prestou a fazer.

Aluna5: *A partir do momento que eu comecei a entender o LOGO eu vi que, para qualquer pessoa, seja adulto, seja criança, é muito mais fácil ele enxergar... do que estar escrito em números e letras. Por exemplo: apareceu um quadrado. Então, qual é a forma do quadrado? [...] se você coloca isso matematicamente no quadro é uma coisa, agora, se você colocar o quadrado de um lado e a forma matemática do outro, e falar: " – Isso é igual a isso"... Então eu acho que faz a pessoa enxergar muito mais rápido.*

Os Alunos destacaram o forte apelo visual do LOGO. A sintaxe e o formalismo estrutural dos programas assumiram uma dimensão gráfica capaz de conduzir à expressão de idéias e sua depuração, características do processo de DERD que norteia a atividade cognitiva de programação de computadores. Segundo eles, a fase de depuração passou a ser facilitada pelo aspecto gráfico do LOGO, que mostrava, muito claramente, onde existia um erro para a correção.

Finalmente, apresentam-se as falas dos entrevistados que apontam a viabilidade do uso do LOGO como uma ferramenta introdutória à programação de computadores. Cabe ressaltar, que essa impressão foi colhida não apenas entre os entrevistados, mas com a maior parte dos alunos que cursaram os "Tópicos Especiais".

Aluno1: *Eu acho que o LOGO tinha que ser apresentado de início [1º/2º períodos] ao invés do PASCAL. Não mudando, assim, a tradição de um ensino de programação. Mas, às vezes, de evoluir mesmo o ensino da programação: uma didática diferente. Porque você ter o contato com uma linguagem que te favorece na criação, na evolução, é melhor que você ter uma dificuldade ao ponto de você ter de passar por ela para conseguir entender. Porque ninguém gosta de entender uma coisa aos trancos e barrancos. [...] Às vezes poderia associar, né? É ter uma introdução com o LOGO e uma apresentação num JAVA. Exemplificando: você mostra o que é uma passagem de parâmetros no LOGO e depois você mostra uma passagem de parâmetro no JAVA.*

O Aluno1 vislumbrou a possibilidade do uso combinado do LOGO com outras linguagens desde o início de seu Curso. Compreendeu-se que o LOGO seria uma ponte entre os conceitos de programação e suas respectivas aplicações com as linguagens tradicionalmente exploradas.

Os argumentos referentes à conexão entre o LOGO e as outras linguagens, apareceram também nos discursos da Alunas 2 e 5. Além disso, essas alunas em conjunto com o Aluno4, destacaram a necessidade de que a Linguagem fosse utilizada no início de suas graduações:

> **Aluna2:** *Só que é igual ao que eu estou te falando, se isso* [contato com o LOGO] *fosse no começo... facilitaria muito a você a visualizar... um comando de repetição dentro de JAVA, nesse sentido. Você aprende a visualizar, você aprende a raciocinar, só de visualizar. "Ah vocês vão fazer um programinha que faz isso, isso e isso". Você já imagina o programa, e como que você vai fazer para ele funcionar. E para a gente não ensinaram assim. Ensinaram os comandos, para depois você imaginar o programa, para depois você voltar e imaginar como é que ia fazer.*
>
> ***Aluna5****: Eu acho que seria muito útil* [uso do LOGO em cursos de SI]*, inclusive se ele tivesse sido aplicado nos períodos iniciais, talvez a gente tivesse tido mais facilidade... Eu acho que faria ver a programação de uma forma mais clara. Eu acho que depois, poderia deixá-lo e começar só nas formas matemática* [outras linguagens de programação].
>
> **Aluno4:** *Então, quando você introduziu a ferramenta SuperLogo, aquilo despertou não só meu interesse como o de muita gente. Pena que não foi você que iniciou o curso. Lá no primeiro, segundo, terceiro período – talvez! Por que a partir dali, fez com que a gente aprendesse muita coisa.*

A possibilidade do uso do LOGO no início do curso de Sistemas de Informação estaria relacionada a uma forma de se propiciar uma adaptação mais suave ao aprendizado de programação. O pensamento de propor o uso da linguagem LOGO junto às já utilizadas reforça o pensamento matético de Papert, discutido na sessão 4.3, possibilitando, por exemplo, conexões entre os conceitos computacionais trabalhados em LOGO e os mesmos nas demais linguagens. Compreendeu-se que uma visão *softer* da programação pode cooperar significantemente para a sua aprendizagem, ampliando as possibilidades de aproveitamento nas UC relacionadas, despertando nos aprendizes o tão cobiçado *empowerment* e preparando-os de uma forma didatizada para a compreensão dos conceitos envolvidos na atividade.

Estabelecer as conexões, no sentido proposto por Papert, entre o LOGO e as outras linguagens é uma forma que ficou entendida como viabilizadora do processo de aprendizagem de programação. As "áreas mais frias" – aquelas essencialmente abstratas dentro da programação – poderiam "ser esquentadas" – tornando-se assim, mais concretas – mesclando-se o uso do LOGO às demais linguagens. Um trabalho que certamente requereria do professor criatividade, dedicação, pesquisa, abertura, compromisso e disposição para encarar os desafios inerentes ao processo.

5.3 – As falas dos professores

Antes de se apresentar a análise de suas falas, é importante ressaltar que os professores que cooperaram com o presente trabalho não correspondem,

necessariamente, àqueles que ministraram aulas para os alunos entrevistados na seção anterior. Elucida-se isto, pois, haja vista o dinamismo do processo educacional, as diretivas administrativas da Universidade Beta e as reivindicações discentes, os profissionais citados pelos alunos podem ter sido substituídos, ou mesmo, terem assumido outras funções fora do ambiente universitário.

A análise que se segue foi estruturada a fim de fornecer elementos para o entendimento das questões que envolvem o professor de programação no processo de ensino-aprendizagem: os professores das unidades curriculares de LPC são conscientes de sua participação na construção da aprendizagem de seus alunos? Estão preparados para o exercício docente? Quais são as suas concepções sobre o exercício pedagógico ligado ao ensino-aprendizagem de programação de computadores?

5.3.1 – Participação na aprendizagem do aluno

O processo de ensino-aprendizagem de programação de computadores envolve a tríade professor-computador-aluno. O computador aparece como um elemento novo da dinâmica educacional: no sentido construcionista, ele é uma ferramenta de expressão de idéias, processos e situações e, especialmente, de resolução de problemas. O processo de aprendizagem se efetiva a partir das interações entre os elementos da tríade: professor e aluno são sujeitos ativos. Do lado docente, qual viria a ser o papel a ser desempenhado visando o sucesso de suas ações?

> **Professor1**: *Bom, primeiro ele tem que conhecer a turma, tem que falar uma linguagem mais próxima das pessoas. Segundo, ele não pode parar de estudar. Ele não pode achar que ele sabe tudo daquela matéria de programação, pois ele pode se confrontar com um aluno que já saiba um pouco mais e que faça perguntas que ele não saiba responder, ou pode se confrontar com aqueles alunos que têm muita dificuldade e ele ter que modificar a sua forma de trabalhar. Tem que ter criatividade, eu acho. Acho que a criatividade é a palavra chave. Porque é fácil dar uma disciplina de caráter normal: a gente prepara as aulas, faz uma apresentação, dá um trabalhinho... é fácil, mas nas disciplinas de programação tem esse viés. Então eu acho que tem que ter criatividade, jogo de cintura...*

O professor demonstrou a necessidade de se conhecer as características de seus alunos. Isso, no sentido de adequar o conteúdo programático aos seus níveis de entendimento, com vistas à efetivação de sua participação na aprendizagem como um facilitador. Seria a busca da identificação da ZDP dos discentes para dentro de seus

limites, atuar. Destacou ainda a importância da constante atualização profissional, o que pode ser associado à dinâmica de renovação das ferramentas de tecnologia. Além disso, demonstrou ter consciência de que, na atualidade, os alunos não são mais totalmente desprovidos de informações sobre os conteúdos tratados academicamente. Este é um indício de que o professor também pode assumir uma postura de aprendizagem em conjunto com os alunos, não perdendo por isso, sua condição de orientador do processo.

No final de sua fala, o Professor1 assume que a docência de programação não é como a de outras disciplinas, por envolver permanente associação entre conceitos e aplicações. Para o exercício de sua atividade, o professor precisa extrapolar a aula transmissiva e se valer de criatividade e "jogo de cintura".

O Professor2 destaca o fator motivacional da ação docente:

> **Professor2:** *Acho que é tentar estimular os alunos. Acho que tentar unir aquilo que a gente está passando ao que eles vivenciam, aos problemas que eles trazem da sua vida. Apesar de isso ser um pouco complexo, tem como você trazer esse lado. Eu acho que quando as coisas vão ficando muito distantes da realidade da pessoa, ela perde um pouco o foco. Então é passar o melhor possível, se apoiar em bons livros e tentar tornar as aulas interessantes para os alunos não perderem aquela vontade de aprender.*

O professor indica a valorização da experiência dos educandos como uma estratégia motivadora da aprendizagem. Entende que os problemas vivenciados pelos próprios alunos devem ser trazidos para a sala de aula. Reconhece que isso não se faz sem esforço, porém não coloca a situação como algo impossível de ser implementado. Além disso, ficou reforçada a questão da atualização permanente de conteúdo e da abordagem pedagógica.

Buscou-se também a discussão das estratégias de condução da aprendizagem dos alunos. Neste sentido, o Professor1 salientou:

> ***Professor1:*** *[...] minhas aulas são todas no laboratório e lá, sempre no começo da aula eu tenho um pouco de teoria. Então: "- Hoje nós vamos ver esse e esse tópico. Quando vocês navegam na WEB, no site tal, lá esse tópico está abordado. E para a gente construir isso, a gente vai precisar disso, disso e disso". Sempre no começo da aula tem um começo de teoria, digamos assim. E ai, depois, a gente já entra na parte prática de como implementar essa teoria.[...] E sempre remeto às linguagens que eles já aprenderam: "- O que vocês fazem em JAVA, aqui [...] nos vamos fazer assim."*

O professor demonstrou que busca não dissociar a teoria de programação da prática efetiva no laboratório e nas situações cotidianas. Num ritmo alternado,

proporciona conexões entre os aspectos teóricos do conteúdo e exemplos de uso acessíveis aos alunos. O processo é seguido por implementações usando a linguagem explorada. O aproveitamento dos conceitos prévios de programação também é valorizado, visando uma melhor adaptação do aluno ao paradigma de programação em questão (imperativo, orientado a objeto, lógico etc.).

O Professor2 assumiu uma postura diferente:

> **Professor2:** *[...] primeiro eu tento fazer uma amizade com o aluno, é claro que mostrando o meu papel e o papel dele. Eu acho que o meu papel é passar o conhecimento e tudo aquilo que eu leio a respeito do assunto. E o dele é estudar e procurar freqüentar as aulas, me questionar, me apertar [...]Agora, você tem que ter certa maleabilidade na forma de lidar. Às vezes, uma turma sente uma dificuldade num conteúdo que outra não sente [...] A gente tem que ser um pouco dinâmico. Não pode ser aquela caixa de Maisena que não muda nunca. [...] Senão, amanhã, passam 10 anos e você está naquela mesma metodologia.*

O modelo transmissivo ficou evidenciado no discurso do Professor2. Entretanto, o mesmo demonstrou ter abertura de espírito para se lançar a outras possibilidades, admitindo que a flexibilidade é indispensável no processo educacional. Ao ser questionado se revia suas estratégias e formas de conduzir as interações em sala de aula, ele afirmou:

> **Professor2:** *Sim, até porque nem tudo é certo para todo mundo. Às vezes, você usa uma tática para passar um conceito para um aluno, que é um pouco mais difícil, você vê que uma turma absorve aquilo bem, outra não. Então você tem que tentar passar aquilo de uma outra forma. Geralmente eu me planejo, me preparo. Agora, se você vê que os alunos tiveram alguma dificuldade, você tem que voltar, não pode deixar a pessoa sem saber. A gente sempre tem que andar com algumas "cartas na manga", às vezes ali, no "tempo de execução"* [da aula] *você tem que rever.*

E malgrado a permanência do modelo transmissivo, agora na idéia de absorção de conteúdos, o professor assume a necessidade de sempre voltar e tentar adequar a forma de abordar um conteúdo.

Nesse sentido, a respeito da revisão de estratégias de ensino, o Professor1 declarou ter o hábito de buscar adequar a cada turma o seu modo de trabalho:

> **Professor1:** *Geralmente, a gente tem um conteúdo programático para cumprir e o ritmo de cada turma é diferente. Essa disciplina, eu já havia lecionado duas vezes. Então, é a terceira vez agora. A turma anterior ela teve um ritmo mais lento. Eu dava mais ênfase à parte teórica, passava um pouco menos de conteúdo, porque eles tinham um pouco mais de deficiência na parte da lógica, na parte básica da linguagem, nas estruturas de repetição. Eu demorei um tempo a mais nesses conteúdos em detrimento de outros do conteúdo programático. Na turma desse semestre, ela já*

tem um pouco mais de base e a gente já está vendo outras funcionalidades. Que a outra turma, inclusive, não viu.

Compreendeu-se que os professores vivem um processo de amadurecimento de suas concepções sobre a docência e que são possuidores de elementos que poderão lhes favorecer o desempenho. Notou-se flexibilidade e desejo de conhecer, aprimorar e aplicar novas formas de interação com vistas à facilitação da aprendizagem, o que abrirá espaço, talvez, para uma participação mais ativa na construção do conhecimento com os alunos.

5.3.2 – Preparo para a docência

Os professores entrevistados possuem formação na área de Ciência da Computação. Ambos continuaram seus estudos na área da tecnologia, sendo que o Professor1 possui uma especialização e exerce a docência há dois anos e meio. Já o Professor2 possui mestrado e exerce a docência há dois anos. Este breve histórico reforça a afirmativa de que os recursos humanos que são selecionados para exercer a profissão de professor das UCs de programação são, em maioria, os bacharéis em Ciência da Computação. Um curso de bacharelado em computação, geralmente, não possui uma fundamentação pedagógica que vise fornecer ao seu alunado elementos para o exercício da docência. Pelo contrário, a ênfase é essencialmente sobre a técnica computacional. Surge, então, o desafio pedagógico para o professor bacharel. Nesse sentido, foi questionado aos entrevistados o que um professor de programação precisaria saber para exercer a docência.

Professor1: *Bom, a parte teórica da linguagem ele tem que saber tudo. Tem que saber os possíveis erros que os alunos vão cometer, ele tem que saber como contorná-los. Eu acho que ele tem que ter vivenciado a parte prática da linguagem: ter feio algum sistema com aquela linguagem, ou feito alguma aplicação. Ele já tem que ter construído alguma coisa naquela linguagem para poder mostrar o lado prático da coisa para os alunos: "- olha meus meninos, sabe quando você está navegando na internet e vem aquele site assim, assim, assim? Por trás está rodando isso aqui que a gente vai ver na aula!".*

O Professor1 evidenciou a necessidade do domínio técnico daquilo que se propõe lecionar. Certamente, esse aspecto é satisfatoriamente trabalhado ao longo de sua formação de graduação e pós-graduação. Entretanto, a fala do docente revela que,

para o exercício do magistério, torna-se necessária uma perspectiva que extrapola o domínio da técnica, ou seja: a mediação da aprendizagem. Corrigir um erro de um programa de computador é diferente de se amparar um aluno durante um momento de erro, quando esse está aprendendo a programar. A função do professor mediador iria além da do bacharel inserido em uma atividade de cunho não pedagógico. O professor destaca ainda a importância da vivência prática naquilo que ele traz para a sala de aula. A contextualização do conteúdo estaria ligada a algo que efetivamente já funciona, que seja de conhecimento dos alunos e que, preferencialmente, tenha sido vivenciada por ele.

O Professor2 também seguiu a mesma linha de raciocínio:

> **Professor2:** *Ele precisa gostar de estudar, estar sempre se atualizando. Porque tudo está constantemente mudando. [...] Saber ser criativo na hora de elaborar provas e exercícios, [...], ter um pouco de amor à profissão: gostar mesmo da área, porque é uma área difícil. Tem que gostar de estudar, ter afinco, um pouco de dedicação.*

No primeiro momento do discurso do professor, o domínio técnico da área foi reforçado, surgindo posteriormente a dimensão pedagógica, a qual estaria ligada a uma forma atrativa para se expor e tentar promover as tarefas e avaliações pertinentes ao processo. A afinidade com a docência de programação foi elucidada, pois a atividade não é vista como um trabalho simplório.

Assim, lecionar programação promoveria o encontro entre a área técnica da computação e uma forma pedagógica para promover a aprendizagem. Por serem bacharéis, os professores sentiriam falta de uma bagagem pedagógica que lhes favorecessem o exercício da docência?

> **Professor1:** *Sinto falta disso. A minha formação é técnica, mais tecnológica, mais de Exatas. Mas quando a gente vai para frente de sala de aula, a gente se depara com situações com comportamentos, que a parte exata, a parte técnica não resolve. Então, eu sinto falta sim! Mais por essas situações. Eu não sei lidar com certos comportamentos dos alunos, ou não sei se a minha forma de lecionar está adequada. Eu sempre pergunto para eles: "- E ai? Vocês entenderam?". Procuro ter esse feedback, mas nem sempre ele é real... Eu já fui aluno e às vezes a gente fala coisas para o professor que ele gosta de ouvir.*

> ***Professor2:*** *Eu acho que o que acontece é que os cursos de graduação são muito focados nas disciplinas específicas então a gente não tem tempo, [...] e essas disciplinas são tidas como complementares nos cursos de Exatas. Então, se você quer ter mais, você tem que ir por sua conta. Quanto mais, melhor. Essa parte de conhecer o aluno, a parte comportamental é importante.*

Em ambos os casos, notou-se a necessidade de uma formação mais específica para o exercício pedagógico. Nesse sentido, as eventuais diferenças entre competência técnica, pedagógica e de gestão da turma aparecem no ambiente de sala de aula. Ficou caracterizada, a necessidade de uma adaptação no fazer docente com vistas ao bom desempenho, sendo isto para o Professor1, um motivo de reflexão em conjunto com seus alunos.

5.3.3 – O processo de ensino-aprendizagem de programação

Nesta última seção da análise, procurou-se identificar o que um professor de programação consegue perceber a respeito do processo de ensino-aprendizagem no qual está inserido, no sentido de indicar quais seriam os elementos importantes e inerentes a esse processo.

> **Professor1:** *Primeiro fator: tem que ter didática do professor. Vou tentar te dar um exemplo: quando eu estava na graduação eu tive um professor que chegava, botava um código fonte numa transparência e explicava linha por linha daquele código fonte. Podia ser bom para alguns, que entendiam aquele código, mas para a grande maioria não era atrativo. [...] Então eu acho que tem que ter uma didática que seja atrativa para o aluno.*

O profissional elucida a necessidade da didatização dos conteúdos. Ao fornecer um exemplo de sua graduação, mostra perceber que nem todo docente tem a clareza de que seu trabalho precisa ser inteligível ao alunado, revelando ainda que uma melhor apresentação/adequação do conteúdo funcionaria como um fator motivacional. O Professor1 continuou a tecer seus comentários:

> **Professor1:** *Segundo ponto: tem que ter infra-estrutura. Não adianta você querer dar uma aula de programação só dentro da sala. Você tem que ir para um laboratório, de preferência com um micro por aluno, se possível com menos pessoas, porque não adianta você querer ir para um laboratório com 40 ou 50 alunos. Você não consegue atingir os mesmos objetivos, se você fosse para esse mesmo laboratório com 15 ou 20 alunos. Penso que é melhor você dar duas aulas para 25 alunos do que dar uma aula para 50, em termos pedagógicos.*

Infra-estrutura é condição necessária ao bom andamento da dinâmica educacional. Um grupo reduzido, com acesso individual às ferramentas de programação também se revelou como desejável para o melhor exercício docente. A mediação se efetivaria de uma maneira mais adequada nestas condições. Finalmente, o professor destacou que:

> **Professor1:** *O aluno tem que ter uma disciplina e ter uma vontade de estudar aquilo muito maior que em outras disciplinas. O aluno da disciplina de programação, se ele não pegar, não sentar, não programar, não errar o código, o compilador ou interpretador não mostrar o código* [de erro] *para ele e ele quebrar a cabeça ele não faz. E ele vai ficando um aluno deficitário. Então, eu penso que os alunos têm que ter uma dedicação especial às disciplinas de programação. Ainda que ele não goste delas.*

Configurou-se na fala do Professor1 a percepção de que a aprendizagem constitui-se nas interações entre professores, alunos e ferramentas. O professor destaca que os alunos precisam se dedicar aos estudos de uma forma disciplinada e sistematizada, o que implica o processo de DERD, destacando a etapa da depuração como fundamental para a consolidação do conhecimento.

De uma forma mais simplificada, também o Professor2 enfatizou o domínio do conteúdo pelo docente e a participação dos alunos como elementos importantes no processo:

> **Professor2:** *Acho que são duas vertentes: O conhecimento do professor e o interesse do aluno. Do professor são os elementos básicos: tem que ter uma formação para aquilo, tem que ser autodidata, estar acompanhando as tendências do mercado, para a gente formar sempre pessoas capacitadas. E dos alunos um pouco de dedicação e criatividade.*

A complexidade do processo de ensino-aprendizagem de programação de computadores foi abordada junto aos entrevistados, questionando o que haveria de tão especial ou de específico nas UCs de programação. Seriam elas diferentes das outras disciplinas?

> **Professor1:** *Existem disciplinas que você apenas precisa transmitir um conteúdo. Você estudou aquilo antes e vai passar aquele conteúdo para o aluno. Se o aluno der aquela decorada, ou então, ler umas quatro ou cinco vezes vai conseguir internalizar aquilo e aquilo vira um aprendizado. Talvez não vire conhecimento, por que tem diferença... Agora, as disciplinas de programação, eu acho que elas requerem raciocínio do aluno e uma certa criatividade do aluno, porque os problemas são distintos. Penso que o professor tem que fazer um esforço de não apenas transmitir aquele conteúdo. Vou voltar ao exemplo da minha graduação: meu professor da graduação simplesmente transmitia o conteúdo. [...] Eu destaco a importância da parte prática e, na medida do possível, o professor acompanhar o aluno nos momentos de erro. "- ah professor, aqui não funcionou!". Dele ir lá e acompanhar isso.*

Em suas considerações, o Professor1 vem caracterizando as inadequações do modelo transmissão de conteúdos, ou do instrucionismo, com as UCs de programação. A mediação durante a prática aparece novamente como um fator preponderante na ação

docente, principalmente no momento do erro: oportunidade viabilizadora da aprendizagem.

O Professor2 assumiu uma postura similar:

> **Professor2***: O ensino de programação tem um conteúdo programático que você tem que seguir, mas não é aquela receita de bolo que você estuda, você lê um texto, tira dele elementos e vai perguntar sobre aquilo. A programação você pode dar uma série de problemas* […] *e tem que ter um raciocínio lógico para resolver aqueles problemas.*

Para o docente, apesar das UCs de programação serem enquadradas dentro de parâmetros curriculares padronizados, o processo de ensino-aprendizagem envolveria uma dinâmica que estaria além da tradicionalmente explorada. Entendeu-se, portanto, ser necessário implementar ações que favorecem a reflexão e a criação de um ambiente de aprendizagem que priorizasse a solução de problemas de forma lógica.

Finalmente, destaca-se que os professores entrevistados declararam o desconhecimento da linguagem LOGO. O Professor1 salientou que um colega de sala, na graduação havia lhe dito que a Linguagem possuía uma tartaruga robótica. Desta forma, os professores assumiram o desconhecimento das potencialidades computacionais e pedagógicas do LOGO.

CONSIDERAÇÕES FINAIS

As TIC são elementos centrais da sociedade moderna. Sua consolidação, enquanto instrumentos transformadores e capacitadores das ações cotidianas, influenciou e continua a influenciar todas as áreas de conhecimento. O convívio com tais tecnologias se impôs, provocando transformações nas formas de se proceder em diversos campos de atividades humanas. Entre tais mudanças, o presente trabalho procurou enfatizar as demandas criadas para a Educação, a qual passou a ser responsável pela formação de profissionais para a atuação junto às TIC. Neste panorama, enfocou-se especificamente o processo de ensino-aprendizagem de programação de computadores, buscando sua melhor compreensão.

A abordagem construcionista proposta por Papert foi encarada como uma maneira de se (re)pensar a educação a partir das TIC, principalmente no que diz respeito a dinâmicas educacionais envolvendo programação de computadores. A linguagem LOGO, desenvolvida por Papert e equipe, representou a concretização de seu pensamento, que vislumbrava a ampliação do acesso a tais tecnologias de uma maneira não excludente e simplificada, porém não menos rigorosa. Enquanto norteador do ambiente de ensino-aprendizagem, o construcionismo e as ferramentas de *software* apresentaram-se como alternativas ao modelo clássico de ensino, procurando fornecer elementos capazes de propiciar uma formação reflexiva, de instigar o debate e o amadurecimento de idéias, de ampliar a capacidade crítica dos educandos, de concretizar a elaboração de produtos do conhecimento (programas de computador, por exemplo) e, o fundamental, possibilitar a grata sensação de se poder fazer algo concreto com programação, compreendendo o modo de se chegar a tal resultado.

No decorrer da realização do curso de “Tópicos Especiais” com alunos formandos do curso de Sistemas de Informação, as impressões apresentadas no Capítulo

5 foram identificadas no comportamento de vários alunos e não apenas nos que foram entrevistados. Durante o curso, a atitude da maioria do grupo frente a atividades propostas era positiva e pró-ativa. Nos encontros, os alunos agrupavam-se espontaneamente e trabalhavam cooperativamente na construção dos programas propostos. As solicitações de intervenções do professor eram em busca de aprimoramentos, novos recursos ou comandos, ou ainda, para pequenos acertos nos programas desenvolvidos. Alguns alunos buscaram aprimorar seus conhecimentos, dedicando parte de seu tempo extraclasse à melhoria de seus programas LOGO. Outros buscaram materiais de apoio na Internet. Após o término do curso, um aluno buscou ajuda para implementar LOGO em uma escola do nível fundamental e também com fins de continuidade de seus estudos em curso de pós-graduação.

De acordo com os dados do perfil da turma, definido antes da realização do curso, dois terços dos alunos declaram-se dedicados nas UCs de programação cursadas anteriormente e a totalidade deles atribuiu grande importância a elas. Apesar disso, a maioria afirmou apresentar baixo conhecimento na área. Os alunos indicaram também um baixo nível motivacional ao longo dos cursos, muito associado à conduta dos professores. Neste sentido, diante do perfil traçado, o melhor retorno obtido com a UC foi ver o *empowerment* dominar os alunos, que puderam experienciar a grata satisfação de construir programas de computador, compreendendo o seu fazer.

Em adição aos depoimentos colhidos, evidencia-se dessa maneira, o potencial da linguagem LOGO enquanto ferramenta de ensino-aprendizagem de programação de computadores no Ensino Superior. Isso parece justificar-se em boa parte pelo forte apelo visual da Linguagem, que se mostrou como uma forma de concretizar abstrações inerentes à atividade de programação. Esta característica possibilitou a visualização das ações propostas nos programas, consolidando uma (re)aprendizagem de fundamentos lógicos e de estruturação em programação. Desta forma, a LOGO se constituiu como uma linguagem viabilizadora do aprendizado e da motivação dos estudantes. Isso no sentido de permitir o acesso ao ato de programar de uma maneira explícita e simplificada, porém, não menos rigorosa que as tradicionalmente exploradas, sendo recomendável seu uso combinado a outras linguagens.

Este trabalho constatou a complexidade própria do processo de ensino-aprendizagem de programação e buscou sua melhor compreensão. Desta forma, procurou-se determinar as impressões de alunos e professores enquanto agentes da dinâmica educacional.

Com a ajuda dos discentes foi possível perceber, que ao se introduzir o computador em um processo educacional, embora existam condições necessárias para instauração de uma nova perspectiva na aprendizagem, o modelo da transmissão de conteúdos predomina como método de ensino. Nesse sentido, a fragmentação dos assuntos e a falta de didatização constituíram-se como elementos bloqueadores da aprendizagem.

Conforme já estabelecido, o perfil do grupo de alunos que cooperaram com o presente trabalho indicou um baixo aproveitamento nas UCs de programação. Percebeu-se entre eles um sentimento de desânimo e mesmo de aversão frente às oportunidades do contato com a programação de computadores. Dessa forma, para eles, a efetivação da aprendizagem foi prejudicada. Dois indicativos colhidos relativos ao problema motivacional nas referidas UCs são a baixa afinidade dos educandos com a área da tecnologia e o desconhecimento dos discentes a respeito do currículo do curso de Sistemas de Informação, o qual tem na programação elemento central de estudos. Em outras palavras: uma escolha mais bem informada levaria os sujeitos a terem maiores condições de alcançar melhores níveis de aprendizado, desempenho e interesse pela área de programação.

A atividade de programação de computadores não é simplista: exige dedicação e envolve raciocínio lógico, capacidade de abstração, concatenação de idéias e conhecimentos prévios. Em geral, todos esses pré-requisitos são pouco explorados e incentivados ao longo da formação de Ensino Fundamental e Médio, os quais, preponderantemente, seguem o modelo transmissivo do conhecimento. Dessa forma, configura-se um cenário de dificuldade para os ingressantes do curso de Sistemas de Informação. Isso não quer dizer que a adoção de uma nova postura frente à aprendizagem não possa ser assumida pelos alunos; ao contrário, isto a eles se impõe.

Da mesma forma, o processo de ensino-aprendizagem de programação exige uma contrapartida dos professores. Compreendeu-se que um dos principais fatores

motivacionais da aprendizagem é a forma de condução das UCs pelos professores. A didatização e contextualização dos conteúdos, a adequação da linguagem utilizada nas exposições, o preparo técnico e pedagógico e a vivência prática na área de programação por parte dos docentes são características que foram entendidas como necessárias para uma melhor mediação da aprendizagem. Indiscutivelmente, a docência de programação é uma atividade que exige o conhecimento técnico e formal dos conteúdos. Ela demanda, igualmente, uma postura pedagógica que estruture e viabilize a aprendizagem dos envolvidos. Revela-se, portanto, um processo plural e que extrapola tecnicismos.

As entrevistas realizadas com dois professores também foram fundamentais para a melhor compreensão do processo de ensino-aprendizagem, uma vez que as interações professor-computador-alunos sintetizam a dinâmica educacional conforme tratada neste trabalho. Nesta tríade, o professor tem uma participação importante e que reflete na aprendizagem dos alunos. Sua inserção em um processo que, como já dito, extrapola o tecnicismo, exige a mediação cooperativa para a promoção da aprendizagem. Além do domínio técnico dos temas, a dimensão pedagógica é um elemento de grande importância no trabalho deste profissional, o que foi explicitado pelos alunos nas entrevistas ao demandarem uma mediação didatizada e contextualizada dos conteúdos. A ausência de uma formação pedagógica configura-se como uma lacuna para os professores, os quais vêm de cursos que enfocam essencialmente a parte técnica da computação. Torna-se, portanto, necessário que o trabalho do professor de programação inclua a capacidade de promover a aprendizagem de uma forma estimulante. Isso no sentido de adequar a técnica da computação a uma dimensão acessível aos discentes.

Finalmente, é importante destacar que a infra-estrutura – computadores, ferramentas de *software*, acesso à internet etc. – é condição necessária à concretização do conjunto de indicativos anteriores. Adcionalmente, turmas com número reduzido de alunos, por exemplo, permitiriam ao profissional responsável pela mediação dos conteúdos uma melhor atuação, ampliando as possibilidades da gerência do aprendizado e um melhor nível de atendimento a cada um dos discentes.

* * *

A origem da linguagem LOGO remonta aos anos 60 do século passado. Fruto de um desejo pessoal de possibilitar um acesso plural à tecnologia, inclusive às crianças, Papert comandou seu desenvolvimento no MIT. No Brasil, cerca de vinte anos depois de seu surgimento, destacou-se o uso da linguagem em frentes de trabalho e pesquisa que incluíram programas de implementação de laboratórios de informática e formação de professores. Dois exemplos são o projeto EDUCOM do Núcleo de Informática na Educação da UNICAMP e o trabalho do LEC-UFRGS (Laboratório de Estudos Cognitivos, da Universidade Federal do Rio Grande do Sul), focado no acompanhamento do aprendizado de crianças em idade escolar, incluindo aquelas com dificuldade de aprendizagem e portadoras de necessidades especiais. Esses trabalhos renderam muitas frentes de pesquisa que fomentaram uma ampla discussão e desdobramentos do trabalho de Papert no Brasil. Recentemente, no entanto, pouco tem sido falado acerca da aplicação da Linguagem LOGO. Apesar de todo o esforço em torno do LOGO, teria a Linguagem sido relegada?

Outro aspecto inegável é que a LOGO é comumente associada ao universo infantil. Esse pensamento acaba por produzir uma distorção frente às potencialidades que a Linguagem oferece, tanto no sentido computacional como no pedagógico. De forma geral, essa idéia preconcebida parece justificar a pouca penetração da linguagem LOGO em ambientes acadêmicos de Nível Superior. Surge então mais um questionamento: a suposição de que LOGO foi concebida em moldes pueris impede, definitivamente, a sua exploração com adultos ou seu desuso é fruto de desconhecimento?

A crítica ao modelo transmissivo de ensino se perpetua no meio educacional. A proximidade da informática à educação abriu possibilidades para uma nova forma de se conceber o processo de aprendizagem. Então, quais as maneiras possíveis de se incorporar, na formação dos professores, a apropriação das TIC? Entre estas, quais são mais adequadas? E ainda: como estão os currículos das licenciaturas em relação às demandas geradas pela cibercultura?

A partir de 2006, o Ministério da Educação (MEC) adotou em seus projetos de inclusão digital o uso do Linux Educacional – atualmente na Versão 3.0 – (figura 6.1), o qual possui diversas ferramentas nativas para a exploração em sala de aula.

Cabe ressaltar que a primeira delas é a *KTurtle*, um ambiente de programação LOGO. Reitera-se aqui a necessidade de se estabelecer projetos que fomentem o desenvolvimento crítico e a implementação de idéias no computador por parte de alunos e professores, conforme materializado na linguagem LOGO. Outra linha de ação diz respeito à formação de professores para o uso da informática na educação. Somados, estes esforços poderão constituir uma abordagem pedagógica alternativa ao modelo transmissivo de conteúdos.

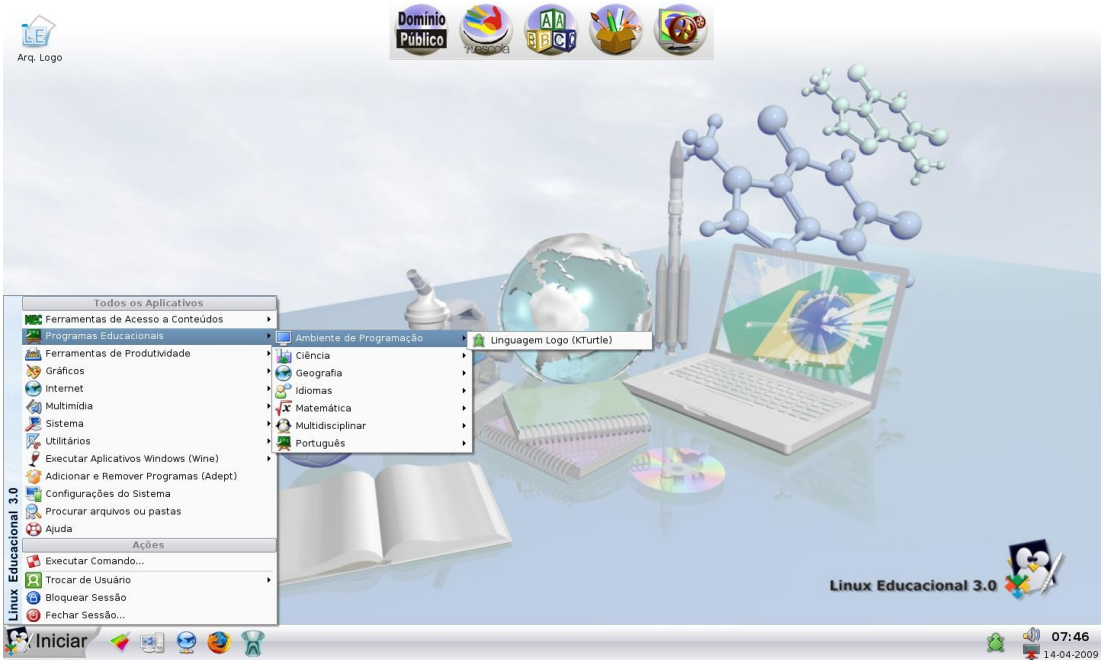

Figura 6.1 – O Linux Educacional 3.0 do MEC com a ferramenta *KTurtle* instalada: Linguagem LOGO.

REFERÊNCIAS BIBLIOGRÁFICAS

ALMEIDA, Maria Elizabeth Bianconcini. *A Informática Educativa na Usina Ciência da UFAL*. Maceió, AL, Anais do II Seninfe, NIES/UFAL, 1991.

_____. LEGO-Logo e Interdisciplinaridade. Porto Alegre, RS. *Anais do VII Congresso Internacional Logo e I Congresso de Informática Educativa do Mercosul*. LEC/UFRGS, 1995.

_____. A Formação de Recursos Humanos em Informática Educativa Propicia a Mudança de Postura do Professor? In Valente J. A. (org), *O Professor no Ambiente Logo: Formação e Atuação*. Campinas, SP, Unicamp/NIED, 1996.

_____. *Informática e Formação de Professores*. Coleção Informática para mudança na Educação. MEC/ SEED/ ProInfo, 1999.

_____. O aprender e a informática: a arte do possível na formação do professor. *Cadernos Informática para a Mudança em Educação*. MEC/SEED/PROINFO, 1999a.

_____. *O computador na escola: contextualizando a formação de professores: Praticar a teoria e refletir a prática.* 2000. Tese de Doutorado em Educação: Currículo. Pontifícia Universidade Católica de São Paulo, São Paulo, 2000.

ALTOÉ, Anair. O trabalho do facilitador no ambiente Logo. In: VALENTE, J.A. *O professor no ambinete Logo: formação e atuação*. Campinas, SP: NIED/UNICAMP, 1996.

ALTOÉ, Anair. *A gênese da informática na educação em um Curso de Pedagogia: a ação e mudança da prática pedagógica*. Tese de Doutorado em Educação: Currículo. Pontifícia Universidade Católica de São Paulo, São Paulo, 2001.

ALTOÉ, Anair; PENATI, Marisa Morales. O Construtivismo e o Construcionismo Fundamentando a Ação docente. In: ALTOÉ, Anair; COSTA, Maria Luiza Furlan; TERUYA, Teresa Kazuko. *Educação e Novas Tecnologias***.** Maringá: Eduem, 2005, p 55-67.

ASCENCIO, Ana Fernanda Gomes. *Lógica de programação com Pascal*. São Paulo: Pearson Education/Makron Books, 1999.

ASCENCIO, Ana Fernanda Gomes.; CAMPOS, Edilene AparecidaVeneruchi de. *Fundamentos da programação de computadores. Algoritmos, Pascal e C/C++. Pascal*. São Paulo: Pratice Hall, 2002.

BARDIN, Laurence. *Análise de conteúdo*. Lisboa: Edições 70, 1977.

BARRELLA, Fernanda Maria Freire; PRADO, Maria Elisabette Brisola Brito. Da Repetição à Recriação: Uma Análise da Formação do Professor para uma Informática na Educação, in Valente, J. A. (org), *O Professor no Ambiente Logo: Formação e Atuação*. Campinas, SP, Unicamp/NIED, 1996.

BARANAUSKAS Maria Cecília Calani. *Procedimento, função, objeto ou lógica?* Linguagens de programação vistas pelos seus paradigmas. In VALENTE, José Armando. (ORG.) Computadores e conhecimento: repensando a educação. Campinas: UNICAMP/NIED, 1998. p.45-63.

BOGDAN, R. C.; BIKLEN. S. K. *Investigação Qualitativa em Educação*. Porto: Porto, 1994

BRASIL. Ministério da Educação. CNE/CES 436/2001: *Cursos Superiores de Tecnologia – Formação de Tecnólogos*. Brasília, 2001, 21p.

BRASIL. Ministério do Planejamento, Orçamento e Gestão. Instituto Brasileiro de Geografia e Estatística: *Pesquisa Anual de Serviços*. Brasília, 2006, 12p. Disponível: <http://www.ibge.gov.br> Acesso em: 13/02/2009.

CASTORINA, José Antônio. O Debate Piaget-Vygotsky: A busca de um critério para sua avaliação, in Piaget Vygotsky: *Novas Contribuições para o Debate*. 2a ed., São Paulo, Ática, 1996.

CHAVES DE CASTRO, *et al*. Utilizando Programação Funcional em Disciplinas Introdutórias de Computação. *XI Workshop de Educação em Computação - WEI 2003*. Campinas, SP, Brasil.

CHEVALLARD, Yves. *La transposición didáctica*:del saber sabio al saber enseñado. Buenos Aires, Aique, 1991.

DECI, E. L.; RYAN, R. M. (1985). *Intrinsic motivation and self-determination in human behavior*. New York: Plenum.

RYAN, R. M., CONNELL, J. P.; DECI, E. L. (1985). A motivational analysis of elfdetermination and self-regulation in education in C. Ames e R. Ames(Orgs.), *Research on motivation in education* (p. 16-31). New York: Academic Press.

DELGADO, C. *et al*. Uma Abordagem Pedagógica para a Iniciação ao Estudo de Algoritmos. *XII Workshop de Educação em Computação (WEI'2004)*. Salvador, BA, Brasil.

DEWEY, John. *Como pensamos: como se relaciona o pensamento reflexivo com o processo educativo: uma reexposição*. São Paulo: Editora Nacional, 1979.

DEWEY John. *Experiência e Educação*. 3a ed., São Paulo, Cia. Ed. Nacional, 1979.

DICIONÁRIO ELETRÔNICO HOUAISS DA LÍNGUA PORTUGUESA. Versão 1.0. Rio de Janeiro: Objetiva, 2001.

FARRER, Henry. *et al*. *Pascal Estruturado*. 2ª.ed. Rio de Janeiro : Guanabara Koogan, 1995.

FERREIRA, Aurélio Buarque de Holanda. *Novo dicionário eletrônico Aurélio versão 5.0*. CD-ROM. Curitiba: Positivo, 2004.

FORBELLONE, A.L.V.; EBERSPÄCHER, H.F.. *Lógica de programação*. São Paulo: Pearson Education/Makron Books, 1999.

FREIRE, Fernanda Maria Pereira; PRADO, Maria Elisabette Brisola Brito. Professores Construcionistas: A Formação em Serviço, in *Anais do VII Congresso Internacional Logo e I Congresso de Informática Educativa do Mercosul*. Porto Alegre, RS, LEC/UFRGS, 1995.

FREIRE, Paulo. *Pedagogia da autonomia: saberes necessários à prática educativa.* São Paulo: Paz e Terra, 1996.

FREIRE, Paulo. *A Educação na Cidade*. São Paulo, SP, 2a ed., Cortez, 1995.

FREIRE, Paulo. *Pedagogia do Oprimido*. 6a ed., Rio de Janeiro, Paz e Terra, 1976.

GUIMARAES, Sueli Édi Rufini; BORUCHOVITCH, Evely. O estilo motivacional do professor e a motivação intrínseca dos estudantes: uma perspectiva da Teoria da Autodeterminação. In *Psicologia Reflexão e Crítica*, Porto Alegre, v. 17, n. 2, 2004. p.143-150. Disponível em<http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0102-79722004000200002&lng=en&nrm=iso>. Acessado em 17/06/2009.

GUIMARAES, Sueli Édi Rufini.; BZUNECK, José Alonso; SANCHES, Samuel Fabre. Psicologia educacional nos cursos de licenciatura: a motivação dos estudantes. In *Psicologia Escolar e Educacional.* Jun. 2002, vol.6, no.1, p.11-19. Disponível em : <http://pepsic.bvs-psi.org.br/scielo.php?script=sci_arttext&pid=S1413-85572002000100002&lng=pt&nrm=iso>. Acessado em 12/06/2009.

HARPER, Babette et al. *Cuidado: escola, desigualdade, domesticação e algumas saídas.* 25.ed. São Paulo: Brasiliense, 1980.

LAKATOS, Eva Maria; MARCONI, Marina de Andrade. *Fundamentos de metodologia científica*. 4. ed. rev. e ampl. São Paulo: Atlas, 2001

LAUDON, Kenneth. C., LAUDON, Jane. P. *Sistemas de informação gerenciais*. 7. ed. São Paulo: Pearson Prentice Hall, 2007.

LÉVY, Pierre. *Cibercultura*. São Paulo: Editora 34, 1999.

LUCENA, Carlos; FUCKS, Hugo. *A educação na era da Internet: professores e aprendizes na WEB*. Rio de Janeiro: Clube do Futuro, 2000.

MACHADO, João Carlos Ribeiro. *O olhar dos alunos e dos professores sobre a informática no curso de Licenciatura em Matemática na UFPA*. 2005. 137p. Dissertação (Mestrado em Educação). Universidade Federal do Pará, Núcleo Pedagógico de Apoio ao Desenvolvimento Científico, 2005.

MALTEMPI, Marcus V. Novas Tecnologias e Construção de Conhecimento: reflexões e perspectivas. In: *Congresso Ibero-Americano de Educação Matemática - CIBEM*, 5., 2005,

Porto, Portugal. Anais do V CIBEM. Porto: APM - Associação de Professores de Matemática de Portugal, 2005. v.1, p.01-10.

MALTEMPI, Marcus V., VALENTE, José Armando. Melhorando e Diversificando a Aprendizagem via Programação de Computadores. In: *International Conference on Engeneering and Computer Education - ICECE 2000*. Proceedings (CDROM). São Paulo, Agosto/2000.

MARCHI, Maria Teresinha Golon. *Informática na educação: uma experiência na formação inicial do professor*. Dissertação de Mestrado. Faculdade de Filosofia, Ciências e Letras de Jandaia do Sul, Jandaia do Sul, 2001.

MARTINS, Sheila Wesley; CORREIA, Luiz Henrique de Andrade. O Logo como Ferramenta Auxiliar Desenvolvimento do Raciocínio Lógico – um Estudo de Caso. *International Conference on Engineering and Computer Education - ICECE 2003*, Santos. Acessado em 28-12-2004. http://www.inf.ufsc.br/~scheila/icece2003.PDF

MORAES, Maria Cândida. *O paradigma educacional emergente*. 11. ed. Campinas: Papirus, 1997.

OLIVEIRA, M. K. de. *Vygostsky: aprendizado e desenvolvimento: um processo sócio-histórico*. São Paulo: Scipione, 1997.

PAPERT, Seymour. *Logo: Computadores e Educação*. Brasiliense, São Paulo, 1985. (Original de 1980)

_____. *Constructionism: A New Opportunity for Elementary Science Education. A proposal to the National Science Foundation.* Massachusetts Institute of Technology, Media Laboratory, Epistemology and Learning Group, Cambridge, Massachusetts. 1986.

_____. "Introduction". In I. Harel (Ed.), *Constructionist Learning*. Cambridge, MA: MIT Media Laboratory. 1990.

_____. *A Máquina das Crianças:* Repensando a Escola na Era da Informática. Ed. rev. Porto Alegre, Artes Médicas, 2008. (Original de 1993)

PEREIRA, J.; ROCHA, J. C.; RAPKIEWICZ, C. E. O Processo de Ensino e Aprendizagem de Algoritmos e Programação: Uma Visão Crítica da Literatura. In: *III Workshop de Educação em Computação e Informática do Estado de Minas Gerais*, 2004, Belo Horizonte, 2004.

PERRENOUD, Philipe. *10 Novas competências para ensinar*. Porto Alegre: ArtMed, 2000.

PETRY, Patrícia Gerent. et al. Estudo Exploratório sobre a Depuração de Programas em Logo, in *Anais do VII Congresso Internacional Logo e I Congresso de Informática Educativa do Mercosul.* Porto Alegre, LEC/UFRGS, 1995.

PETRY, Patrícia Gerent. *Um sistema para o ensino e aprendizagem de algoritmos utilizando um companheiro de aprendizado colaborativo*. 82p. Dissertação de mestrado (Ciência da Computação). Universidade Federal de Santa Catarina, 2005.

PIAGET, Jean. *A Epistemologia Genética*. Ed. Vozes, Petrópolis, RJ, 1972.

PIAGET, Jean. *Fazer e Compreender*. São Paulo, Editora Melhoramentos e Editora da Universidade de São Paulo, 1978.

PORTO ALEGRE, Laíze Márcia. *Utilização das tecnologias da informação e da comunicação, na prática docente, numa instituição de ensino tecnológico.* Campinas-SP.2004. 235p. Tese (Doutorado em Educação). Universidade Estadual de Campinas. 2005.

PRADO, Maria Elisabette Brisola Brito. Logo no curso de magistério: o conflito entre abordagens educacionais. In: VALENTE, José Armando (Org.). *Computadores e conhecimento: repensando a educação*. Campinas, SP: Gráfica Central da UNICAMP, 1993.

PRADO, Maria Elisabette Brisola Brito. *(Re)visitando o construtivismo para a formação do professor reflexivo*. 1998. Disponível em: <http://www.c5.cl/leinvestiga/actas/ribie98/239.html>. Acesso em: 14/07/2008.

PRADO, Maria Elisabette Brisola Brito. *O Uso do Computador na Formação do Professor: um enfoque reflexivo da prática pedagógica*. Coleção Informática para a Mudança na Educação. Brasília: MEC/SED, 1999.

QUARTIERO, E. M. *Da máquina de ensinar à máquina de aprender*. Pesquisas em Tecnologia educacional. Revista Vertentes, São João del-Rei, n.29, p.51-62, jan./jun. 2007.

RIBEIRO, J. G. *O Ambiente Logo como Elemento Facilitador na Reflexão Pedagógica sobre a Prática Educativa*, Monografia do Curso de Especialização em Informática na Educação. Maceió, NIES/UFAL, 1994.

RODRIGUES JÚNIOR, Methanias Colaço. *Como Ensinar Programação?* Informática - Boletim Informativo Ano I n° 01, ULBRA. Canoas, RS, Brasil, 2002.

SALOMON, G.; PERKINS, D.N.; GLOBERSON, T. Coparticipando en el conocimiento: la ampliación de la inteligencia humana con las tecnologias inteligentes. *Comunicación, lenguage y educación*, n.13.6-22, 1992.

_____. *Educational psychology and technology*: A matter of reciprocal relations Teachers College Record, V.100, n.2, p.222-241, 1998.

_____. Cogniciones distribuidas. Consideraciones psicológicas y educativas. Buenos Aires: Amorrortu, 2001.

SANTOS, Rodrigo Pereira; COSTA, Heitor Augustus Xavier. Análise de Metodologias e Ambientes de Ensino para Algoritmos, Estruturas de Dados e Programação aos iniciantes em Computação e Informática. *INFOCOMP – Journal of Computer Science*, Lavras/MG – Brasil, v. 5, n. 1, p. 41-50, 2006.

SCHÖN, Donald A. Formar Professores como Profissionais Reflexivos, in Nóvoa, A. (org.), *Os Professores e sua Formação*. Lisboa, Portugal, Dom Quixote, 1992.

SCHULTZ, Max Rúbens de Oliveira. *Metodologias para Ensino de Lógica de Programação de Computadores*. Monografia de Especialização (Ciência da Computação). Universidade Federal de Santa Catarina (UFSC), Florianópolis, SC, Brasil. 2003, 69p

SOUZA, Jesus Maria; FINO, Carlos Nogueira. *As TIC abrindo caminho a um novo paradigma educacional. Educação e Cultura Contemporânea.* Rio de Janeiro, v.5, n.10, p.11-26,jan./jun. 2008.

TIMOTEO, Dênio Mariz; BRASILEIRO, Francisco Vilar. A Formação e a Prática do Profissional de Programação:Discussão de Estratégias para a Aprendizagem Continuada. In: *Congresso da SBC, 1998, Belo Horizonte - MG*. *Anais do Workshop de Educação em Informática*, 1998. v. 3.

TURKLEY, Sherry. *The Second Self: Computers and the Human Spirit.* Simon and Schuster, New York, 1984.

VALENTE, José Armando. Diferentes usos do computador na educação. In: *Computadores e conhecimento: repensando a educação.* 1ª ed. Campinas, NIED-Unicamp, 1993.

_____. Por que o Computador na Educação. In: Valente, J. A. (org.) *Computadores e Conhecimento: Repensando a Educação*. Campinas, Gráfica da Unicamp, 1993b.

_____.Análise dos diferentes tipos de softwares usados na Educação. In: Valete, J.A. (org). *O computador na sociedade do conhecimento*. Ed. Campinas: UNICAMP/NIED, 1999.

_____. Mudanças na sociedade, mudanças na Educação: o fazer e o compreender. In: Valete, J.A. (org). *O computador na sociedade do conhecimento*. Ed. Campinas: UNICAMP/NIED, 1999a.

_____. A espiral da aprendizagem e as tecnologias da informação e comunicação: repensando conceitos. In JOLY, M. C. (Ed.) *Tecnologia no ensino: implicações para a aprendizagem*. São Paulo: Casa do Psicólogo Editora, 2002a, p. 15-37.

_____. Aspectos críticos das tecnologias nos ambientes educacionais e nas escolas. *Educação e Cultura Contemporânea: Revista do Programa de Pós-Graduação em Educação da Universidade Estácio de Sá.* Rio de Janeiro, v.2, n.3, p.11-28, 2005.

VYGOTSKY, Lev Semenovich. *A formação social da mente: o desenvolvimento dos processos psicológicos superiores*. Martins Fontes, São Paulo, 1984.

ZEICHNNER, Kenneth M. *A formação Reflexiva de professores*: *Idéias e Práticas*. Lisboa, EDUCA,1993.

ANEXO 1

QUESTIONÁRIO DIAGNÓSTICO

1) Como você situa seu conhecimento sobre "Introdução à Lógica" e "Fundamentos de Programação I"?

a) Excelente b) Bom c) Mediano d)Sofrível e)Péssimo

2) Como futuro profissional da área de Sistemas de Informação, você considera as unidades curriculares "Introdução à Lógica" e "Fundamentos de Programação I":

a) Indispensáveis b) Muito importantes c) Importantes d) De baixa importância e) Dispensáveis

3) Quanto a sua motivação durante os cursos de "Introdução à Lógica" e "Fundamentos de Programação I", você se enquadraria como:

a) Desmotivado b) Indiferente c) Motivado d) Altamente motivado

4) A que você atribui seu grau de motivação? (marque quantas alternativas quiser)

a) Ao professor b) A estar fazendo o que eu realmente gosto c) Às Ferramentas de hardware e software
d) Outra opção: _____

5) Para você, qual é a importância do envolvimento do professor com a condução do aprendizado nas disciplinas de "Introdução à Lógica" e "Fundamentos de Programação I"?

a) Irrelevante b) Importante c) Fundamental

6) Qual é a importância do seu envolvimento no aprendizado dessas disciplinas?

a) Irrelevante b) Importante c) Fundamental

7) Qual é a importância ferramental (Hardware e Software) para o aprendizado nas disciplinas de "Introdução à Lógica" e "Fundamentos de Programação I"?

a) Irrelevante b) Importante c) Fundamental

8) Sobre a sua dedicação nestas disciplinas:

a) Fui muito dedicado (freqüentava e participava das aulas, estudava a teoria, (re)fazia exercícios e buscava outras fontes para aprimorar e expandir os meus conhecimentos e trabalhei em conjunto com meus companheiros de estudo).

b) Fui dedicado (freqüentava as aulas, estudava a teoria, (re)fazia exercícios e trabalhei em conjunto com meus companheiros de estudo).

c) Fui um aluno mediano (me ative a freqüentar as aulas, a estudar para as avaliações e trabalhos).

d) Fui um aluno com baixa dedicação.

9) Quanto ao meu aproveitamento nas disciplinas de "Introdução à Lógica" e "Fundamentos de Programação de Computadores":

a) Excelente b) Bom c) Regular d) Sofrível e) Péssimo, porque

10) Quais são as suas expectativas sobre este curso de "Tópicos Especiais" em Sistemas de Informação?

ANEXO 2

QUESTÕES DAS ENTREVISTAS COM OS ALUNOS

01) FALE UM POUCO DE SUA TRAJETÓRIA NO CURSO DE GRADUAÇÃO.

02) FALE AGORA SOBRE SEU CURSO. (...) VOCÊ GOSTOU OU NÃO? POR QUÊ?

03) PRETENDE SE ESPECIALIZAR, CONTINUAR OS ESTUDOS? (...) EM QUE ÁREA? (...) POR QUÊ?

AS UC´S DE PROGRAMAÇÃO E O PROCESSO DERD

04) FALE UM POUCO DO SEU PERCURSO NAS DISCIPLINAS DE PROGRAMAÇÃO.

05) O QUE SE ESPERA DE UM ALUNO PARA A APRENDIZAGEM DE LINGUAGENS DE PROGRAMAÇÃO?

06) QUAL O SEU GRAU DE ENVOLVIMENTO NAS DISCIPLINAS DE PROGRAMAÇÃO? (...) POR QUÊ? QUALIFIQUE, ENTÃO, O SEU ENVOLVIMENTO EM UMA ESCALA DE 0 A 10.

07) QUANTO À MOTIVAÇÃO DURANTE OS CURSOS DE PROGRAMAÇÃO, QUE VOCÊ TEM A DIZER?

08) A QUE FATORES VOCÊ ASSOCIARIA O GRAU DE MOTIVAÇÃO PARA UM CURSO DE PROGRAMAÇÃO?

09) COMO VOCÊ AVALIA A ATUAÇÃO DOS PROFESSORES DOS CURSOS DE PROGRAMAÇÃO? (...) POR QUÊ? QUALIFIQUE, ENTÃO, ESSA ATUAÇÃO NUMA ESCALA DE 0 A 10.

10) A FORMA DE CONDUÇÃO DAS UNIDADES CURRICULARES DE PROGRAMAÇÃO É SIMILAR AS OUTRAS DISCIPLINAS DE SEU CURSO, OU REQUEREM UM PROCESSO DE ENSINO APRENDIZADO DIFERENCIADO? (...) COMO ASSIM? (...) POR QUÊ?

11) QUAL O PAPEL (RELEVÂNCIA) DAS UNIDADES CURRICULARES DE PROGRAMAÇÃO NO CURSO DE SISTEMAS DE INFORMAÇÃO?

12) O QUE VOCÊ ESPERA DE UM PROFESSOR DENTRO DO PROCESSO DE ENSINO APRENDIZAGEM NAS DISCIPLINAS DE PROGRAMAÇÃO? (...) QUAIS AS CARACTERÍSTICAS DESEJÁVEIS PARA ESSE PROFISSIONAL?

13) QUAIS AS LP'S QUE VOCÊ MAIS SE FAMILIARIZOU? POR QUÊ?

14) FAÇA UMA CRÍTICA OU COMENTÁRIO ESPECÍFICO A ALGUMAS DESSAS FERRAMENTAS?

15) QUAL É, NO SEU ENTENDIMENTO, A IMPORTÂNCIA DO PROCESSO DERD NA ATIVIDADE DE PROGRAMAÇÃO?

16) QUANDO E COMO TOMOU CONSCIÊNCIA DO PROCESSO DERD?

17) APRENDER A PROGRAMAR AJUDA COMO UMA FORMA DE "PENSAR" EM OUTRAS DISCIPLINAS? POR QUÊ?

DO CONTATO COM O LOGO

18) VOCÊ JÁ CONHECIA O LOGO ANTES DA DISCIPLINA DE TÓPICOS ESPECIAIS II?

19) QUAIS SÃO AS SUAS IMPRESSÕES SOBRE O LOGO?

20) O LOGO PODE SER ÚTIL NO PROCESSO DE ENSINO-APRENDIZADO DE LINGUAGENS DE PROGRAMAÇÃO? (...) COMO ASSIM?

21) O LOGO É UMA LINGUAGEM DE PROGRAMAÇÃO PARA CRIANÇAS? (...) POR QUE? (...) COMO ASSIM?
22) QUAL O PONTO FORTE DO LOGO?

23) QUAL O PONTO FRACO DO LOGO?

24) A SUA VISÃO SOBRE O APRENDIZADO DE PROGRAMAÇÃO MUDOU ALGUMA COISA APÓS O CONTATO COM O LOGO? (...) COMO ? (...) POR QUÊ? (...)

25) QUAL SERIA A MELHOR POSIÇÃO PARA O USO DO “LOGO” DENTRO DOS CURSOS QUE ENVOLVEM PROGRAMAÇÃO?

26) POR QUÊ SERÁ QUE EM SEU CURSO O “LOGO” SÓ FOI EXPLORADO NO FINAL?

27) A CONDUTA DO PROFISSIONAL QUE MEDIA O APRENDIZADO COM O LOGO PRECISA SER DIFERENCIADA? EM QUE ASPECTOS? DE QUE MANEIRA?

QUESTÕES DAS ENTREVISTAS COM PROFESSORES

QUESTÕES GERAIS

01) QUAL O SEU TEMPO DE FORMAÇÃO, TEMPO DE FORMADO, TEMPO DE MAGISTÉRIO?

02) QUAIS SÃO OS ELEMENTOS QUE VOCÊ JULGA IMPORTANTES NO PROCESSO DE ENSINO-APRENDIZAGEM DE PROGRAMAÇÃO? POR QUÊ?

03) O QUE O ENSINO DE PROGRAMAÇÃO TERIA DE ESPECIAL, DE ESPECÍFICO? NO QUE ELE É OU DEVERIA SER DIFERENTE DO DE OUTRAS DISCIPLINAS?

SOBRE AS LINGUAGENS DE PROGRAMAÇÃO

04) QUAL A ORDEM DE LINGUAGENS DE PROGRAMAÇÃO VOCÊ ACHA MAIS ADEQUADA PARA OS ALUNOS APRENDEREM?

05) POR QUE TAL LINGUAGEM COMO A PRIMEIRA?

SOBRE O EXERCÍCIO DA PROFISSÃO DOCENTE E A PRÁTICA PEDAGÓGICA

06) QUAL O PAPEL DO PROFESSOR NAS DISCIPLINAS DE PROGRAMAÇÃO DE COMPUTADORES? O QUE ELE DEVE FAZER PARA QUE O TRABALHO SEJA BEM SUCEDIDO?

07) O QUE O PROFESSOR PRECISA SABER PARA DAR AULA DE PROGRAMAÇÃO DE COMPUTADORES?

08) DURANTE O SEU CURSO DE GRADUAÇÃO OU EM PÓS-GRADUAÇÃO VOCÊ TEVE ALGUMA MATÉRIA DE NATUREZA PEDAGÓGICA? ISSO É RELEVANTE, TE AJUDA, VOCÊ SENTE FALTA?

09) QUAIS SÃO SUAS PRINCIPAIS ESTRATÉGIAS DE ENSINO?

10) VOCÊ COSTUMA REVER SUAS ESTRATÉGIAS DE ENSINO? COMO VOCÊ COSTUMA FAZER?

SOBRE O LOGO

11) VOCÊ CONHECE O LOGO?

12) SE CONHECE, QUAIS SÃO AS SUAS IMPRESSÕES SOBRE O MESMO? ELE PARTICIPA EM SUAS DISCIPLINAS?