UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Campus de Ilha Solteira

Desenvolvimento de uma ferramenta computacional de apoio ao ensino de sistemas eletrônicos digitais

Marcelo Luis Murari

Ilha Solteira – SP

Desenvolvimento de uma ferramenta computacional de apoio ao ensino de sistemas eletrônicos digitais

Marcelo Luis Murari

Orientador: ***Prof. Dr. Norian Marranghello***

Dissertação apresentada à Faculdade de Engenharia de Ilha Solteira da Universidade Estadual Paulista, *Campus* de Ilha Solteira, para obtenção do título de Mestre em Engenharia Elétrica.

UNESP – Ilha Solteira
Abril de 2008

Agradecimentos

O primeiro agradecimento dedico, honrosamente, a Deus, meu guia supremo.

Em segundo lugar, agradeço a um outro guia, ao meu orientador Norian Marranghello, que, tão pacientemente e bondosamente, conduziu-me ao fim da luta, transformando cicatrizes em luzes.

Agradeço, também, aos meus pais, guias dos quais recebi, amorosamente, "broncas" e conselhos; ensinamentos, que fizeram de mim, uma pessoa vencedora.

A minha noiva, Ana Paula, pela dedicação, companheirismo e amor maior.

A minha irmã, Nidia, pelo apoio e compreensão a mim desprendidos.

Aos meus amigos, Álvaro, Henrique, André, Alexandre, Cecílio e Jussara, que, com apoio e companheirismo, tornaram-se meus eternos companheiros de jornada.

A cada professor, tanto de Ilha quanto de Rio Preto, que me acompanharam e me conduziram desde o início até o fim desta caminhada, ensinando-me o verdadeiro valor do aprender.

Aos professores de estatística, Elen e Clinton, que, gentil e prestativamente, ajudaram-me com números e gráficos.

Às instituições, FEF – Fundação Educacional de Fernandópolis, UNIJALES – Centro Universitário de Jales, UEMG – Universidade do Estado de Minas Gerais, que, gentilmente, cederam-me o espaço para a realização de testes e comprovação de resultados.

Aos *campi* da UNESP, Ilha e Rio Preto, locais nos quais consegui realizar antigos sonhos e idealizar novos objetivos.

"A cooperação existe entre todas as coisas criadas." – C. Torres Pastorino

Um muito obrigado a todos.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

EaD: Ensino a Distância.

Abed: Associação Brasileira de Educação a Distância

SACI: Sistema Avançado de Comunicações Interdisciplinares

LDB: Lei de Diretrizes e Bases

CALDIS: *Computer-Aided Learning of Digital System*

IDE: *Integrated Development Environment*

VCL: *Visual Component Library*

GUI: *Graphic User Interface*

JVM: *Java Virtual Machine*

OO: Orientação a Objetos

ANSI: *American National Standard Institute*

4GT: Técnicas de Quarta Geração

JSP: *Java Server Page*

Resumo

Neste trabalho é descrito um ambiente de apoio ao aprendizado em sistemas digitais. Ele foi desenvolvido com o objetivo de auxiliar o processo ensino-aprendizagem em Sistemas Digitais, por meio da implementação da ferramenta, intitulada CALDIS *(Computer-Aided Learning of Digital Systems)*, que dinamize esse processo. Nessa ferramenta, alunos e professores interagem por meio de trocas de informações referentes aos assuntos abordados em sala de aula, mediante a leitura de textos de apoio, a simulação de dados e a visualização dos resultados dos dados simulados. Após o desenvolvimento da ferramenta, verificou-se que tal ambiente, no tocante às contribuições ao ensino de Sistemas Digitais, tornou-se um recurso didático-pedagógico eficaz para a compreensão da teoria pretendida. Também, foi possível vislumbrar a viabilidade do emprego desta ferramenta em outras áreas do ensino, não se restringindo, portanto, a Sistemas Digitais nem a Ciências Exatas.

Abstract

This work describes a supporting environment for digital systems learning. It was developed with the aim of helping the teaching-learning process of Digital Systems through the implementation of a tool, which is named CALDIS *(Computer-Aided Learning of Digital Systems)*, to make this process more dynamic. With this tool, students and teachers interact by exchanging information that refers to the subjects seen in class, using support texts, data simulation and the visualization of the simulated data outcome. After its implementation it has been verified that such an environment, as for its contribution to Digital Systems learning, has become an effective didactic-pedagogic resource towards the understanding of the intended theory. It has also been possible to wonder the applicability of this tool in other teaching-learning areas of knowledge, thus, not restricting them to Digital Systems nor to Exact Sciences.

Introdução

"O Ensino a Distância (EaD), também chamado de Teleducação, em sua forma embrionária e empírica, é conhecido desde o século XIX, mas somente nas últimas décadas assumiu *status* que o coloca no cume das atenções pedagógicas de um número cada vez maior de países." (ARAÚJO & MALTEZ, 2003).

Segundo as autoras Suely Trevisan Araújo & Maria Gil Lopes Maltez (2003),

> "o ensino a distância surgiu da necessidade do preparo profissional e cultural de milhões de pessoas que, por vários motivos, não podiam freqüentar um estabelecimento de ensino presencial e evoluiu com as tecnologias disponíveis em cada momento histórico, as quais influenciam o ambiente educativo e a sociedade." (ARAÚJO & MALTEZ, 2003).

Inicialmente na Grécia e, depois, em Roma, existia uma rede de comunicação na qual as cartas comunicando informações científicas inauguraram uma nova era na arte de ensinar. Citando Lobo Neto (1995), Araújo & Maltez (2003), em seu artigo "Educação a Distância: Retrospectiva Histórica" informam que um primeiro marco do ensino a distância foi o anúncio publicado na Gazeta de Boston, no dia 20 de março de 1728, pelo professor de taquigrafia Cauleb Phillips: "Toda pessoa da região, desejosa de aprender esta arte, pode receber em sua casa várias lições semanalmente e ser perfeitamente instruída, como as pessoas que vivem em Boston."

Nessa perspectiva, em 1833, um anúncio publicado na Suécia já se referia ao ensino por correspondência e na Inglaterra, em 1840, Isaac Pitman sintetizou os princípios da taquigrafia em cartões postais que trocava com seus alunos.

No entanto, o desenvolvimento de uma ação institucionalizada de ensino a distância teve efetivamente início a partir da metade do século XIX.

Na década de 40, o EaD despontou no cenário brasileiro com os cursos por correspondência oferecidos pelos pioneiros institutos Monitor e Universal Brasileiro.

Segundo, Frederic Michael Litto (2004 apud GUIMARÃES, 2004), presidente da Associação Brasileira de Educação a Distância (Abed), "na década de 1970 o Brasil deu um grande salto no *e-learning* com o Telecurso e o projeto SACI (Sistema Avançado de Comunicações Interdisciplinares), mas recuou na década seguinte, devido à descontinuidade governamental. Apenas as iniciativas privadas de cursos por correspondência sobreviveram."

Em "Educação a Distância: em que ponto estamos?" Aldé (2003) faz uma breve retrospectiva do EaD no Brasil:

> "... de 1970 até hoje, a TV se disseminou - e com ela os telecursos -, o videocassete surgiu - multiplicando o acesso aos conteúdos -, depois o fax, e, finalmente, há pouquíssimo tempo, o computador e a Internet se consolidaram como meios educativos. Na história do EaD, ao contrário de outras áreas, cada nova tecnologia não descarta as anteriores, pelo contrário: os diversos recursos complementam-se. O rádio continua sendo o principal meio de comunicação em regiões de difícil deslocamento. O telefone aproxima alunos e professores, facilitando a solução de dúvidas que não podem esperar. E ainda não inventaram nada mais concreto e permanente que o material impresso, que pode ser levado para qualquer lugar, não depende de eletricidade nem linha telefônica. Tudo isso ao lado das mais modernas invenções de interatividade que a tecnologia digital propicia: *e-mail*, fórum, *chat*, teleconferência etc."

Segundo o presidente da Abed (Associação Brasileira de Educação a Distância), a retomada do processo, no Brasil, deu-se na década de 1990 quando o então presidente Fernando Henrique Cardoso criou a Secretaria de Ensino a Distância no MEC; "a área de informática foi mais priorizada que a educação no governo FHC, o foco foi redirecionado no governo Lula", pontua Litto (2004 apud GUIMARÃES, 2004).

No Brasil, o EaD é uma alternativa à formação regular desde 1996, quando passou a ser regulamentado pela Lei de Diretrizes e Bases (LDB).

Segundo Araújo & Maltez (2003), atualmente o ensino não-presencial mobiliza os meios pedagógicos de quase todo o mundo, tanto em nações industrializadas como em países em desenvolvimento. Cursos novos e mais complexos são desenvolvidos, tanto no âmbito dos sistemas de ensino formal quanto nas áreas de treinamento profissional.

Ainda sob a ótica dessas duas autoras, "hoje, cada vez mais é também usado (o EaD) em programas que complementam outras formas tradicionais, face a face, de interação e é visto por muitos, como uma modalidade de ensino alternativo que pode substituir parte do sistema regular de ensino presencial."

Nessa perspectiva, a CALDIS, uma ferramenta de apoio ao aprendizado em Sistemas Digitais, foi desenvolvida para ser um instrumento de interação entre professor-aluno, numa relação de troca do conhecimento da área de Engenharia, em Sistemas Digitais.

A presente dissertação será organizada em cinco capítulos respectivamente intitulados - As relações do processo ensino-aprendizagem, Comparação de linguagens de programação, Engenharia de *Software,* Descrição do trabalho, Resultados obtidos – seguidos do tópico "Considerações Finais", que trará uma avaliação dos aspectos pedagógicos e técnicos do desenvolvimento da CALDIS.

Capítulo 1. As relações do processo ensino-aprendizagem

Neste capítulo são apresentadas considerações acerca das relações do processo ensino-aprendizagem quanto à problematização dos sujeitos - professor, aluno - e das ações - ensinar, aprender - envolvidos nesse processo educacional. Ainda nessa perspectiva, são definidos estilos de aprendizagem que atendem aos mais variados níveis de desenvolvimento de diferentes alunos, com o intuito de investigar as capacidades desses alunos para identificar as exigências que a CALDIS deve atender enquanto uma ferramenta significativa de apoio à aprendizagem.

1.1. Processo Ensino-Aprendizagem

No tocante às considerações acerca do processo ensino-aprendizagem, foi realizada uma revisão bibliográfica sobre esse assunto e foram apresentados os pressupostos teóricos de Bigge (1977) e de Vasconcellos (1995).

Segundo Bigge (1977), as situações de ensino-aprendizagem classificam-se de acordo com o lugar que ocupam num *continuum* que vai desde os modos de operar "sem pensar" até os que envolvem "pensar". Esse *continuum*, no entanto, segundo esse autor, é dividido em quatro classes amplas: nível de memória, nível de compreensão, nível de desenvolvimento da autonomia e nível de reflexão.

O nível de memória, caracterizado por meio de um tipo de aprendizagem que se restringe a um simples armazenamento de fatos na memória, é o que implica em menos pensamento; o de reflexão, sustentado por um exame crítico do conhecimento a ser adquirido à luz da constatação que o apóia e que pode ser comprovada e das conclusões a que dirige, é o que implica em mais pensamento. No nível de compreensão, a aprendizagem dá-se por meio da acomodação da informação nova às estruturas do pensamento; no nível do desenvolvimento autônomo, o aprender advém do

desenvolvimento da ampla consciência intuitiva de si mesmo que cada aluno apresenta e da expressão artística da sua auto-avaliação.

Dessa forma, no nível de desenvolvimento autônomo, o professor exerce pouca liderança, direção, coerção, prescrição ou imposição sobre o pensamento ou o comportamento dos alunos e, portanto, o ensino é centrado no aluno, que é encorajado, por meio do estímulo ao desenvolvimento de sua criatividade, a liberar seus impulsos naturais e seus sentimentos. Conseqüentemente, o desenvolvimento intelectual ocorre naturalmente e ao estudante dever-se-ia permitir que escolhesse as atividades escolares a serem por ele desenvolvidas em uma atmosfera amplamente permissiva.

Apesar dos produtos no nível de compreensão e no nível de reflexão serem compreensões ou *insights,* no nível de reflexão é acrescentada uma maior habilidade do aluno em adquirir maior compreensão pelo pensamento. Devido à natureza e aos resultados do processo do ensino no nível reflexivo, esse nível é mais adequado para professores e escolas comprometidos com um relacionamento democrático entre professor e aluno.

Nessa perspectiva, Vasconcellos (1995) fundamenta que o planejamento do processo ensino-aprendizagem "deve partir da realidade concreta tanto dos *sujeitos*, quanto do *objeto* de conhecimento e do *contexto* em que se dá a ação pedagógica". Assim, a primeira atitude deve ser a do professor, enquanto mediador do processo ensino-aprendizagem, no sentido de investigar e conhecer a realidade com a qual vai trabalhar (vivência dos alunos, condições da escola, necessidades da comunidade, etc.), além do conhecimento que tem sobre o objeto de estudo e a realidade global do processo educacional. Para tanto, a sistematização e a avaliação do trabalho do ano anterior é uma importante referência norteadora a ser problematizada para o desenvolvimento do planejamento do ano letivo a ser iniciado.

Uma vez compreendida a importância da realização do planejamento, algumas questões devem ser fundamentadas para a sua elaboração: conhecimento dos sujeitos; conhecimento dos determinantes; conhecimento do objeto do conhecimento e conhecimento do contexto.

1.1.1. Os Sujeitos

Os sujeitos participantes do processo ensino-aprendizagem são o professor e o aluno:

- O professor deve estar consciente de *seu* projeto, reconhecendo-se nos diferentes pontos de vista: **humano** - traços de firmeza de caráter, respeito, tolerância; **ético** - princípios, parâmetros, coerência, senso de justiça, preocupação com o bem comum; **intelectual** - capacidade de refletir, capacidade de rever os pontos de vista, inteligência no trato com a realidade, apreender seu movimento, ir além do senso comum; **profissional –** desenvolvimento de competências, domínio da matéria e da metodologia de trabalho, segurança nos conceitos e técnicas, interesse, ânimo no que faz, preparo das aulas, atualização (VASCONCELLOS, 1995).
- O aluno deve ser reconhecido a partir da realidade na qual está inserido, de suas necessidades, de seus interesses, de seu nível de desenvolvimento, de seu quadro de investigações, de suas experiências anteriores.

Numa perspectiva social, o professor deve conhecer os determinantes da vivência dos alunos, visto que o universo cultural, social, político e econômico dos mesmos estruturam-se por elementos comuns: situações de vida muito semelhantes, mesma condição social, acesso aos mesmos meios de comunicação e de informação, etc. (VASCONCELLOS, 1995).

Para que o processo ensino-aprendizagem seja significativo, faz-se necessário que o professor conheça o objeto a ser trabalhado, de forma a apropriar-se dele. Segundo Vasconcellos (1995), esse conhecimento dá-se em dois níveis: o objeto de conhecimento em si, e as representações que os alunos têm dele.

O primeiro nível refere-se ao domínio que o professor apresenta sobre o conteúdo nos aspectos de sua concepção, de seu desenvolvimento e de sua face interdisciplinar; o segundo alude ao conhecimento prévio do aluno em relação ao objeto de estudo.

Na prática pedagógica, o conhecimento prévio deve sempre ser considerado como ponto de partida para a aquisição de um conhecimento sistematizado e crítico, visto que o fator mais decisivo na aprendizagem é o que diz respeito àquilo que o aluno sabe sobre o objeto de estudo.

Como uma última questão a ser problematizada destaca-se a necessidade do professor reconhecer que a aprendizagem é desenvolvida numa determinada realidade, num determinado contexto que abrange desde a sala de aula até a sociedade como um todo e que está inserido num sistema sócio-político-econômico-cultural. Dessa forma, o professor deve tomar consciência das pressões a que está sujeito, pois somente assim ele conseguirá desenvolver um trabalho crítico e consciente que adentre as estruturas sociais.

Tomar o senso comum como objeto de análise, a visão dos alunos como ponto de partida, é considerar que o seu pensamento e a sua fala relevam aspectos metodológicos a serem considerados no trabalho conjunto deles com os professores nas situações de ensino, com a finalidade de romper com a evidência do senso comum mediante um novo código de leitura da realidade, construindo um novo universo conceitual e aceitando o desafio de confrontar e transformar o senso comum e de

transformar-se nesse processo. A ciência e o conhecimento, enquanto síntese sobre o mundo por intermédio do processo de pesquisa, só terão sentido à medida que possibilitarem a compreensão e a transformação desse mesmo mundo.

1.2. Estilos de Aprendizagens

Para o estabelecimento do estilo de aprendizagem a ser adotado pela CALDIS foram revisados os estudos de Kolb (1984), Dunn e Dunn (1987).

Partindo de sua definição de estilo de aprendizagem como "um processo derivado das interações entre o indivíduo e o ambiente", David Kolb (1984) inferiu que as pessoas, frutos de diferentes ambientes e, portanto, detentoras de condições e experiências distintas e de bagagens hereditárias peculiares, desenvolvem estilos pessoais de aprendizagem que evidenciam, priorizam algumas habilidades em relação a outras.

Nessa mesma linha de pesquisa, Dunn e Dunn (1987) postulam a existência de várias condições que afetam a aprendizagem, as quais podem ser agrupadas em quatro categorias, a saber: a) *ambientais*, que dizem respeito à luminosidade, temperatura, som e maior ou menor formalidade na situação de aprendizagem; b) *emocionais*, concernentes à responsabilidade, motivação, persistência e necessidade ou não de estruturas mais formais para aprender; c) *sociais*, referentes à preferência de aprender sozinho ou em grupo e d) *físicas*, que incluem, além das modalidades preferenciais de atenção (visual, auditiva, sinestésica ou mista), horários de maior rendimento e necessidade de movimentação e alimentação como condição facilitadora da aprendizagem.

Nessa perspectiva, Kolb defende que as pessoas aprendem por meio de suas interações com o ambiente e das escolhas envolvidas nessas interações e que, conseqüentemente, cada uma desenvolve características próprias de pensamento,

adotando diferentes estilos de decisão e resolução de problemas como respostas aos desafios impostos pelo ambiente.

Desde 1971, Kolb dedicou seus estudos ao conhecimento de como os indivíduos concentram-se, absorvem, processam e assimilam informações novas. Seu ponto de partida foi a elaboração de um modelo de representação do modo como as pessoas aprendem determinado *modelo de aprendizagem vivencial* por enfatizar o papel significativo da experiência, que influencia e modifica as situações, que, por sua vez, conduzem a novas experiências, para o processo de aprendizagem.

Para Kolb, o processo de aprendizagem é representado por um ciclo quadrifásico que exige quatro habilidades:

a) Experiência concreta: capacidade de se envolver, por meio dos sentimentos e do uso dos sentidos, completa, aberta e imparcialmente em novas experiências;
b) Observação reflexiva: observação e reconhecimento de diferentes perspectivas e pontos de vista; reflexão acerca das experiências;
c) Conceitualização abstrata: elaboração de conceitos lógicos que ofereçam fundamentação teórica às observações;
d) Experimentação ativa: emprego das teorias na resolução de problemas e na tomada de decisões.

Partindo da premissa que o processo de aprendizagem necessita de diferentes perspectivas opostas (ação e reflexão, envolvimento concreto e distanciamento analítico, etc.), Kolb, em 1976, propôs um *inventário de estilo de aprendizagem* (*Learning Style Inventory - LSI*) para mensurar a proeminência individual dos alunos referente a cada uma das habilidades. Nesse inventário, é enfatizado um modelo estrutural de aprendizagem centrado na pessoa, que demanda duas dimensões

fundamentais para o processo de aprendizagem, cada qual admitindo duas orientações elementares em oposição dialética:

- Dimensão concreto/abstrato – abstração sobre a concretude;
- Dimensão ativo-reflexiva – experimentação sobre a reflexão.

No inventário de estilo-aprendizagem do modelo vivencial de Kolb, que descreve o modo como as pessoas aprendem e lidam com as situações do dia-a-dia em sua vida, desenvolvido em 1981 e revisado em 1985, são apresentadas doze sentenças com quatro alternativas, que representam a opinião dos alunos a respeito de qual situação seria aprendida mais facilmente e qual seria mais difícil de ser aprendida. Portanto, o objetivo é que os alunos classifiquem as sentenças a partir de uma progressão gradual dos níveis de aprendizagem de acordo com o seu estilo de aprendizagem.

O respondente ordena as quatro palavras em cada fileira na seqüência que melhor descreva sua forma de aprender. Para aferir a ênfase relativa em cada orientação de aprendizagem, quatro escores são calculados pela soma do número de ordenação dos seis termos mais representativos da respectiva orientação. Em acréscimo, dois escores compostos são determinados: um indica a diferença de ênfase entre abstração e concretude; o outro indica a diferença de ênfase entre ação e reflexão. O espaço bidimensional formado pelos dois escores compostos permite identificar quatro estilos individuais - resultantes da combinação de duas orientações predominantes, uma em cada dimensão -, que foram assim denominados: Acomodador ou Adaptativo, Divergente, Assimilador e Convergente:

- Acomodador ou Adaptador – Aprendem por meio da experimentação ativa e da experiência concreta, sobretudo, realizando tarefas desafiadoras. Adaptam-se facilmente às circunstâncias imediatas; solucionam seus problemas por ensaio e

erro e, na maioria das vezes, agem movidos pela emoção e pela intuição. Ainda necessitam do outro para a busca de informações.

- Assimilador – Fazendo uso de seu raciocínio intuitivo e de sua habilidade para operar idéias lógicas por meio da abstração e da fundamentação teórica, aprendem por observação reflexiva e conceituação abstrata. Muitas vezes, tornam-se incapazes de resolver problemas práticos que exijam a simples aplicação do conhecimento adquirido.
- Divergente – Aprendem por meio da experiência concreta e da observação reflexiva, contemplando as situações sob diferentes perspectivas e em suas múltiplas relações. Adaptam-se bem a situações que exigem novas e inusitadas idéias, valendo-se de sua criatividade e de sua habilidade de criar novas alternativas para a resolução de problemas. Entretanto, podem apresentar dificuldades para a tomada de decisões.
- Convergente – Valendo-se do raciocínio hipotético dedutivo, bem como da conceituação abstrata e da experimentação ativa, resolvem com facilidade e habilidade, situações que apresentam uma única solução. A aplicação prática de conhecimentos aliada a sua capacidade de solucionar problemas e de tomar decisões são suas características marcantes. No entanto, a sua agilidade e praticidade podem induzí-los a soluções e decisões errôneas.

Neste capítulo, foi verificado que a ferramenta de ajuda ao aprendizado em Sistemas Digitais - CALDIS foi elaborada com o intuito de atender a todo e qualquer usuário, independente do estilo de aprendizagem apresentado pelo mesmo, visto que cada um aprende e formula conceitos de acordo com suas condições ambientais, emocionais, sociais e físicas, adequando o conhecimento oferecido a sua capacidade de aprender, abstrair e formular, a sua experiência concreta, a seu estilo de aprendizagem.

A CALDIS objetiva que o estudante esteja envolvido (experiência concreta), veja e escute (observação reflexiva), crie idéias (conceituação abstrata) e tome decisões (experiência ativa) - um processo ativo de aprendizagem.

Capítulo 2. Comparação de Linguagens de Programação

Considerando que um programa deve obedecer a pontos cruciais como eficiência e portabilidade, no presente capítulo são discutidas as linguagens de programação Java, C e Delphi com o intuito de determinar qual dentre as três oferece melhores recursos de desenvolvimento para a CALDIS, cuja aplicabilidade fundamenta-se pela sua caracterização enquanto um recurso pedagógico para a sistematização e automação do ensino de Engenharia.

Num primeiro momento, é apresentada uma breve conceituação de linguagens de programação. Nos tópicos seguintes, as linguagens Java, C e Delphi são discutidas. Posteriormente, sob a ótica do critério de validade e pertinência de suas características de desenvolvimento, é justificada a escolha da linguagem Java para a concepção e implementação da CALDIS.

2.1. Linguagens de Programação

Partindo dos pressupostos teóricos determinados por Sebesta (2003), foi verificado que os estudos dos conceitos de linguagens de programação são fundamentados por alguns benefícios potenciais:

- Aumento da capacidade de expressar idéias. O poder expressivo da linguagem em que os pensamentos são comunicados influencia a profundidade da capacidade intelectual;
- Maior embasamento para escolha de linguagens apropriadas. A familiaridade com os recursos particulares das diferentes linguagens de programação permite ao programador uma escolha fundamentada nas reais funções e aplicabilidades dessas linguagens;
- Capacidade aumentada para aprender novas linguagens. A aquisição de uma completa compreensão dos conceitos fundamentais das linguagens torna mais

fácil a visualização da integração dos conceitos ao projeto da linguagem aprendida;

- Entender melhor a importância da implementação. A compreensão das questões de implementação possibilita o entendimento do porque das linguagens serem projetadas de uma certa maneira, levando a capacidade de usar uma linguagem de modo mais inteligente, dispondo de todas as suas potencialidades;
- Aumento da capacidade de projetar novas linguagens. O conhecimento minucioso da projeção, caracterização e aplicação de cada linguagem pode significar um impulso na projeção de novas linguagens;
- Avanço global da computação. Uma visão global da computação, bem como dos conceitos das linguagens de programação podem determinar a popularização de uma linguagem em desuso da outra, visto que esse conhecimento possibilita ao programador a identificação das reais potencialidades das linguagens.

Ainda no propósito de examinar cuidadosamente os conceitos fundamentais das várias construções e das capacidades das linguagens de programação, fez-se necessário o estabelecimento de algumas considerações acerca dos critérios - legibilidade (simplicidade global, ortogonalidade, sintaxe); escrita (abstração) de avaliação da linguagem.

A simplicidade global de uma linguagem de programação afeta fortemente sua legibilidade - facilidade com que os programas podem ser lidos e compreendidos. Entretanto, a complexidade, a multiplicidade de recursos - mais de uma maneira de realizar uma operação particular - e a sobrecarga - existência de mais de um significado para um único símbolo - são características que comprometem essa simplicidade, visto que os programadores, na maioria das vezes, preocupam-se em aprender unicamente os recursos básicos da linguagem a ser utilizada.

A ortogonalidade - integração entre os conceitos, o grau de interação entre diferentes conceitos, e como eles podem ser combinados de maneira consistente; por exemplo, quando uma *string* não pode ser passada como parâmetro, os conceitos de *string* e parâmetro não podem interagir, e, portanto, há falta de ortogonalidade - está estreitamente relacionada à simplicidade: quanto mais ortogonalidade em um projeto de uma linguagem, menos exceções às regras da linguagem exigirão, o que significa um grau mais elevado de regularidade no projeto, o que torna a linguagem mais fácil de ser aprendida, lida e entendida.

Uma outra característica significativa à legibilidade dos programas é a sintaxe ou a forma dos elementos de uma linguagem. A seguir, são apresentados três exemplos de opções de projeto sintático que afetam a legibilidade:

- Formas identificadoras: restrição dos identificadores a tamanhos muito pequenos que prejudicam a legibilidade.
- Palavras especiais: a aparência do programa e, desse modo, a sua legibilidade são fortemente influenciadas pelas formas das palavras especiais de uma linguagem.
- Forma e significado: projeção de instruções, a fim de que sua aparência indique, pelo menos parcialmente, sua finalidade, é um auxílio evidente para a legibilidade. A semântica, ou o significado, deve seguir diretamente da sintaxe ou da forma.

Fazendo referência à escrita, a abstração apresentou-se como um importante critério de avaliação a ser considerado. O grau de abstração permitido por uma linguagem de programação e a naturalidade de sua expressão são, por conseguinte, muito importantes para sua capacidade de escrita. Esse critério dá-se em duas categorias distintas: processo e dados; a abstração do processo é o uso de um subprograma para

implementar um algoritmo de classificação exigido diversas vezes em um programa e a de dados é o uso de vetores e números inteiros para implementar um algoritmo binário.

Na seqüência desse estudo, foram apresentadas discussões acerca das três linguagens em questão:

2.1.1. Linguagem Java

A linguagem Java foi projetada para ser utilizada no desenvolvimento de aplicações que consumam o mínimo de recursos do sistema e que possam ser executadas em diferentes plataformas de *hardware* e *software.* Os aplicativos em Java são compilados em um código de *bytes* independente de arquitetura.

Java é uma linguagem de programação de alto nível com as seguintes características:

- **Simples:** A simplicidade da Java deve-se ao fato de não possuir sobrecarga de operadores, aritmética de ponteiros e memória alocada dinamicamente, visto que essas memórias são gerenciadas pela própria linguagem.
- **Arquitetura neutra, portável:** Por possuir uma neutralidade em relação à arquitetura do computador, os programas em Java são portáveis. Derivado de uma natureza distribuída de cliente/servidor, um importante recurso da Java é o suporte a clientes e servidores em configurações heterogêneas de redes.
- **Compilada e Interpretada***:* A compilação dos programas da Java é feita em formato binário de código de *bytes*, que são interpretados por um ambiente de execução da Java específico da plataforma utilizada. Portanto, é ao mesmo tempo compilada e interpretada. Assim, permite a tradução de um programa Java para um código intermediário chamado de *bytecode*. Os *bytecodes* são independentes de arquiteturas. Com o interpretador, cada instrução *bytecode* é

executada no computador. A compilação, em oposição à interpretação que acontece cada vez que o programa é executado, ocorre apenas uma vez.

- **Orientada a objetos:** As linguagens orientadas a objetos oferecem muitas vantagens sobre as linguagens procedurais tradicionais. Como os objetos encapsulam dados e funções relacionados em unidades coesas, é fácil localizar dependências de dados, isolar efeitos de alterações e realizar outras atividades de manutenção, e o mais importante, as linguagens OO facilitam a reutilização da estrutura de um programa.
- **Distribuída:** A linguagem Java realiza a distribuição de informações para compartilhamento e trabalho conjunto. Felizmente, para os programadores em Java, há uma biblioteca de procedimentos incluída nos códigos-fonte e de distribuição binária do Java. Isso facilita aos programadores o acesso remoto às informações.
- **Alto desempenho:** No desenvolvimento de sistemas implementados em Java, a interpretação de objetos de códigos de *bytes* proporciona desempenho aceitável. Entretanto, outras circunstâncias exigem desempenhos mais altos. A Java concilia essas duas situações oferecendo a tradução dos códigos de *bytes* para o código de máquina nativo em tempo de execução.
- ***Multithreaded*:** Na Java, os objetos binários de códigos de *bytes* são formados por seqüências de execuções múltiplas e simultâneas. Essas seqüências são denominadas contextos de execução ou processos leves. Ela oferece suporte no nível de linguagem para multitarefa, resultando em uma abordagem de programação mais poderosa e de múltiplas faces.
- **Robusta:** A Java possui uma robustez que permite uma maior confiabilidade, ou seja, a maior parte da verificação de tipos de dados é realizada em tempo de

compilação, e não em tempo de execução. Isso evita muitos erros e condições aleatórias nos aplicativos.

- **Segura:** Os aplicativos Java apresentam garantia de resistência contra vírus, pois não são capazes de acessar uma área temporária de armazenamento de dados que permite acesso aleatório, tais como pilhas ou memória do sistema. A autenticação do usuário, na Java, é implementada com um método de chave pública de criptografia. Isso impede, de maneira eficaz, que *hackers* e *crakers* examinem informações protegidas, como nomes e senhas de contas.
- **Dinâmica:** Os programas Java adaptam-se aos ambientes computacionais mutantes que são acessados dinamicamente e vendidos ou distribuídos separadamente dos aplicativos que deles dependem. Isso permite que os programadores tirem total proveito da orientação a objetos.

Ribeiro-Junior (2003) esboçou a compilação e a interpretação de programas Java.

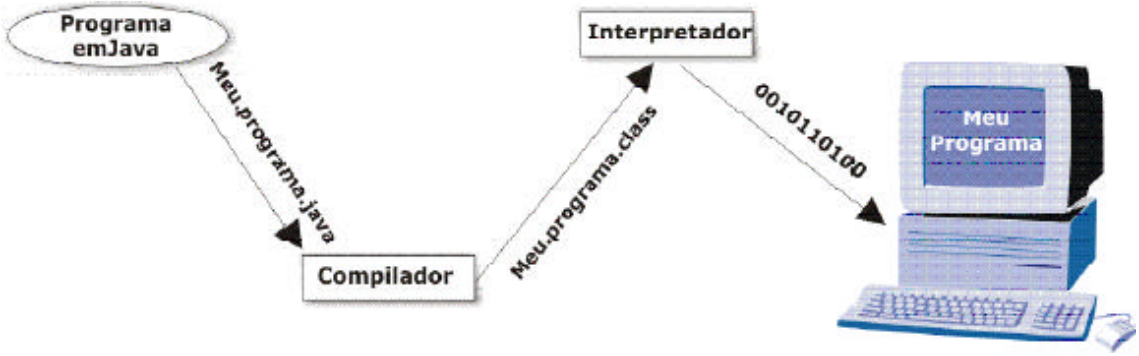

Figura 1 – Compilação e Interpretação de Programas Java.
Reprodução: Ribeiro-Junior (2003)

A figura 1 ilustra o processo de compilação e interpretação de programas Java. Pode-se compilar os programas Java em qualquer plataforma que possua um compilador Java. O produto da compilação (*bytecodes*) pode então ser executado em qualquer

implementação da Máquina Virtual Java. Portanto, o mesmo programa Java pode ser executado em qualquer sistema operacional sem a necessidade de recompilação (figura 2).

"O aspecto da utilização da Java em multiplataforma é muito importante, porque os programadores não necessitam ficar preocupados em saber em qual máquina o programa será executado, uma vez que um mesmo programa pode ser usado num PC, num Mac ou em um computador de grande porte. É muito melhor para uma empresa desenvolver um *software* que possa ser executado em "qualquer lugar" independente da máquina do cliente." (FURGERI, 2002).

Figura 2 – Execução de Programa em Java em Diversos Sistemas Operacionais.
Reprodução: Ribeiro-Junior (2003)

"O grande diferencial de Java em relação às outras linguagens de programação se refere ao fato de que ela foi concebida originalmente para ser usada no ambiente da *World Wide Web*. Nos últimos cinco anos, a grande maioria das linguagens tem buscado se adaptar a essa nova realidade e necessidade, entretanto a Java tem se destacado até o momento." (FURGERI, 2002).

2.1.2. Linguagem C

C é uma linguagem de programação de finalidade geral que permite economia de expressão, modernos fluxos de controle e estruturas de dados e um rico conjunto de operadores. Segundo Kernighan e Ritchie (1989), C não é uma linguagem de "muito alto nível", nem "grande" e nem específica para uma área de aplicação particular. Mas, sua falta de restrições e sua generalidade a tornam mais conveniente e eficaz para muitas tarefas do que linguagens de nível maior.

Enquanto uma linguagem de programação de nível médio, o C apresenta as seguintes características:

- **Estruturada:** Permite a compartimentalização do código e dos dados. Trata-se da habilidade de seccionar e esconder do resto do programa todas as informações necessárias para se realizar uma tarefa específica.
- **Portabilidade:** Caracteriza-se com uma linguagem portável, por possuir neutralidade em relação a diversas arquiteturas de computadores e de sistemas operacionais. Com a padronização feita pelo ANSI, tornou-se ainda mais portável.
- **Modularidade:** Estrutura-se por meio da divisão de sistemas em unidades conceituais - blocos de códigos - que encapsulam informações relacionadas.
- **Recursos de baixo nível:** Apesar de utilizar construções de alto nível, trabalha com a manipulação de *bits*, *bytes* e endereços, os recursos mais básicos para a funcionalidade de um computador.
- **Geração de código eficiente:** Opera com a geração de códigos executáveis compactos e rápidos devido à utilização da função *C Standard Library* (Biblioteca Padrão do C), cujas funções, que estão em formato relocável, transformam endereços relativos (endereços de memória das várias instruções

em código de máquina que não estão absolutamente definidos) em endereços realmente usados.

- **Simplicidade:** Possui uma sintaxe bastante estruturada e flexível, tornando sua programação bastante simplificada.
- **Facilidade de uso:** Apresenta *interface* simples e clara, comandos não-ambíguos e poucas funções intrínsecas.
- **Apontadores e vetores:** Um apontador, que consiste em uma variável que contém o endereço de outra variável, torna-se essencial por ser o único modo de se conseguir fazer passagem por referência. Os vetores são agrupamentos de objetos na memória que se caracterizam como uma matriz unidimensional, cujos elementos pertencem ao mesmo tipo de dado.
- **Reutilização:** O *software* desenvolvido pode ser reutilizado para novas aplicações.
- **Extensibilidade:** Facilidade de introduzir novas funções e adaptar o *software* a mudanças nas especificações.
- **Compilação separada:** Permite que um programa seja dividido em muitos arquivos e que cada arquivo seja compilado separadamente. Uma vez que todos os arquivos estejam compilados, eles são linkeditados com qualquer rotina de biblioteca, para formar um código-objeto completo.

A figura 3 descreve o fluxo de um compilador C: o código (Fonte C) e o pacote de dados do sistema e do usuário (respectivamente, *Headers* do sistema e *Headers* do usuário) são processados, compilados e montados, gerando um Arquivo Objeto que é ligado a Arquivos Objeto do Usuário e a Bibliotecas do Sistema, gerando um executável.

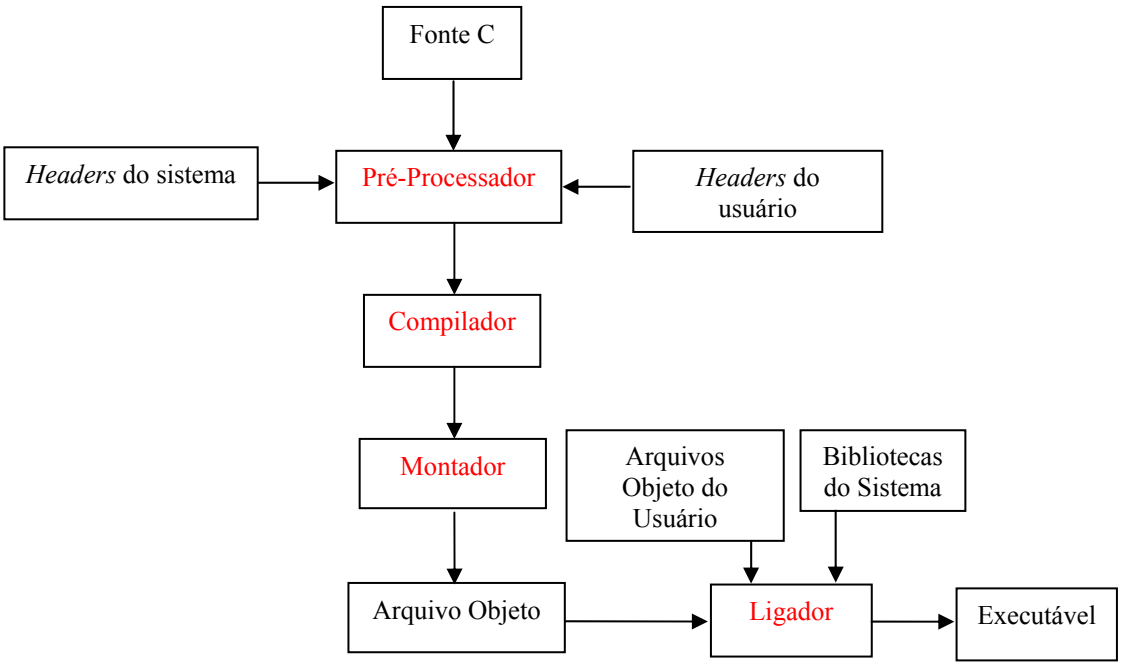

Figura 3 – Fluxo do Compilador C

"Uma particularidade interessante no programa C é seu aspecto modular e funcional em que o próprio programa principal é uma função. Esta forma de apresentação da linguagem facilita o desenvolvimento de programas, pois permite o emprego de formas estruturadas e modulares encontradas em outras linguagens". (SCHILDT, 1996).

2.1.3. Linguagem Delphi

As linguagens visuais implantaram recursos estratégicos radicalmente novos na área de programação. Frente às dificuldades encontradas para a implementação de programas, principalmente daqueles que exigiam *interface* gráfica, alguns programadores perceberam que muitas coisas que eram difíceis de ser feitas, como construir janelas, menus ou botões, podiam ser feitas sempre da mesma forma. Estes programadores, habituados a colecionar sub-rotinas de utilização geral, passaram a encapsular algumas dessas rotinas em uma espécie de "objeto" pronto para ser usado. Assim, a percepção de que vários destes objetos podiam simplesmente ser desenhados na tela como se desenha um retângulo ou outra figura qualquer deu origem ao Delphi,

uma ferramenta para desenvolvimento de *software*, que possui um ambiente de desenvolvimento integrado e um compilador.

Nessa perspectiva, o Delphi, linguagem visual, possui as seguintes características:

- **Compilador de código nativo de alto desempenho:** Apresenta um compilador rápido e otimizado que gera executáveis sem a utilização de bibliotecas para a distribuição de aplicações, que permite a criação de componentes nativos, ou seja, permite que se faça extensões do próprio Delphi e que possibilita a análise e reutilização de códigos de terceiros.
- **Orientada a objetos:** Utiliza o *Object Pascal*, que oferece a facilidade de programação em quarta geração de alto nível com o desempenho da terceira geração.
- **Construtor visual de interface com o usuário:** Possui um IDE - *Integrated Development Environment* - Ambiente de Desenvolvimento Integrado que permite a criação de aplicações Cliente/Servidor.
- ***Two-way*:** Cada componente visual é implementado por meio de um conjunto de linhas de código e esses dois elementos, componente visual e linha de código, estão intimamente relacionados, de tal forma que uma alteração em qualquer um deles se reflete no outro.
- **Biblioteca de componentes visuais** *(VCL – Visual Component Library):* É formada por objetos reutilizáveis e por objetos padrões de *interface* com o usuário, e também, por gerenciamento de dados, por gráficos e multimídia e por gerenciamento de arquivos.
- **Depurador gráfico:** Permite encontrar e eliminar *bugs* em seu código.

- **Gerenciador de projetos** *(Project Manager):* Oferece uma visualização de todos os formulários e unidades de um determinado projeto e um mecanismo conveniente para gerenciar projetos.
- **Gerenciador de relatórios** *(ReportSmith):* Possibilita a geração de relatórios para desenvolvedores que precisem criar relatórios com grandes volumes de dados.
- **Servidor Local baseado em SQL:** Permite o desenvolvimento *off-line* econômico com um *engine* (máquina) SQL de alta performance compatível com ANSI que proporciona acessibilidade a outros servidores.
- **Manipulação de exceção** *(Exception Handing):* Possui mecanismos especiais para manipulação de exceção que permite a criação de aplicações mais robustas.
- **Modelos** *(Templates):* Permite que o desenvolvedor construa formulários e projetos padronizados e armazene-os em um repositório, para serem reutilizados em trabalhos posteriores.
- **Interfaces gráficas do usuário** *(GUIs - Graphic User Interface):* Os usuários podem passar mais tempo procurando dominar o aplicativo e menos tempo se preocupando com o as funções de cada seqüência de teclas nos menus e caixas de diálogo. Assim, erros de programação não ocorrem com tanta freqüência e, se ocorrem, são bem mais fáceis de descobrir e corrigir.

Segundo Cornell e Strain (1995) os programas nas linguagens de programação convencionais seguem de cima para baixo. Para as linguagens de programação mais antigas, a execução tem início na primeira linha e se desloca com o fluxo do programa para partes diferentes, conforme necessário. Um programa Delphi funciona de forma completamente diferente. O núcleo de um programa Delphi é um conjunto de trechos de código independentes que são ativados por eventos que eles foram ensinados a

reconhecer. Essa é uma mudança fundamental. Em lugar de um programador projetar um programa para o que ele deseja, é o usuário quem está com o controle.

2.2. A Escolha da Linguagem

Após o estudo dessas três linguagens, foi realizada a escolha da linguagem para o desenvolvimento da CALDIS.

A arte de escolher uma linguagem consiste, logo de início, em decidir quais são os seus requisitos e a sua importância relativa, uma vez que provavelmente será impossível satisfazer todos igualmente bem (com uma única linguagem)... linguagens disponíveis devem ser medidas em relação a uma lista de requisitos... (MEEK; HEATH, 1980).

Nessa perspectiva, dentre as linguagens Java, C e Delphi, optou-se pela Java para o desenvolvimento da ferramenta de apoio ao aprendizado em Sistemas Digitais – CALDIS, pois a Java é uma linguagem que pode ser executada em diversas plataformas de *hardware* e *software*, característica relevante e indispensável se considerada a possível demanda pela execução da CALDIS nas mais variadas plataformas.

Essa linguagem destacou-se, também, por ser compilada e interpretada. A compilação dos programas é feita pela tradução dos mesmos para os *bytecodes* que, durante a interpretação, são executados no computador. Essa particularidade garante à CALDIS a redução do tempo de execução dessa ferramenta, visto que a compilação dos programas, em oposição à interpretação, ocorre uma única vez.

A linguagem Java, por ser uma linguagem orientada a objetos, oferece, ainda, a possibilidade de usar como base um objeto já disponível (herança), característica que, também, economiza muito tempo no desenvolvimento de objetos e minimiza a ocorrência de erros. Além disso, esses objetos, que são compostos por seqüências múltiplas e simultâneas, oferecem suporte no nível de linguagem para multitarefa o que

confere a Java uma abordagem de programação de múltiplas faces. Nesses parâmetros, o desenvolvimento da CALDIS não só se torna mais rápido, mas também se apodera das inúmeras faces da Java.

Aliadas a essas características, a confiabilidade e a segurança oferecidas por Java proporcionam à CALDIS por meio, respectivamente, da verificação de tipos de dados em tempo de compilação e da autenticação do usuário a proteção das informações nele disponibilizadas.

As características da linguagem Java aqui apresentadas são indispensáveis à concepção e implementação da CALDIS, uma ferramenta centrada em um aprendizado construído por meio da interação aluno/professor.

Capítulo 3. Engenharia de *Software*

Fritz Bauer (1969 apud PRESSMAN, 1995) propôs uma primeira definição de engenharia de *software*, na primeira grande conferência sobre o assunto:

> "O estabelecimento e uso de sólidos princípios de engenharia para que se possa obter economicamente um *software* que seja confiável e que funcione eficientemente em máquinas reais."

Seja sob a óptica da definição oferecida por Bauer (1969), ou ainda, sob a das mais variadas definições, a consulta aos pressupostos teóricos e às aplicações práticas ofertadas pela área da engenharia de *software* é tida como uma exigência no processo de desenvolvimento de *software*.

Em "Engenharia de *Software*", Pressman (1995) fundamenta que essa ciência, área responsável pelo planejamento, desenvolvimento e manutenção dos *softwares,* abrange um conjunto de três elementos fundamentais - métodos, ferramentas e procedimentos.

Planejamento e estimativa de projeto, análise de requisitos de *software* e de sistemas, projeto da estrutura de dados, codificação, teste e manutenção e, muitas vezes, uma notação gráfica ou orientada à linguagem e um conjunto de critérios para a instauração e a manutenção da qualidade do *software* foram os métodos empregados na elaboração da CALDIS.

O desenvolvimento da CALDIS partiu da necessidade sentida por professores e alunos da área de Engenharia de se criar um ambiente de apoio ao ensino de Sistemas Digitais. Mediante tal necessidade, esse ambiente começou a ser planejado com vistas a atender às exigências de pessoas atuantes nessa área do conhecimento.

Realizado o planejamento e estabelecidas as estimativas da CALDIS, foi feita uma análise de requisitos e de sistemas para que se tornasse possível ao grupo de

desenvolvedores da CALDIS especificar a função e o desempenho da mesma e estabelecer quais seriam as restrições de projeto que a CALDIS deveria enfrentar. Assim, ficou estabelecido que a CALDIS seria criada com o intuito de ser uma ferramenta pedagógica que otimizasse a participação de professores e alunos no processo ensino-aprendizagem de Sistemas Digitais, sob a única exigência de um possível usuário possuir, instalado em sua máquina, a ferramenta interativa de cálculos MATLAB, desenvolvida pela *MathWorks*, e o dispositivo JVM (*Java Virtual Machine*), desenvolvido pela *Sun Microsystems*.

O projeto de estrutura de dados da CALDIS que prima por uma descrição procedimental do *software* será detalhado no capítulo 4.

Considerando que

> "as representações do projeto devem ser convertidas numa linguagem artificial (a qual pode ser uma linguagem de programação convencional ou uma linguagem não procedimental usada no contexto do paradigma 4GT) que resulte em instruções que possam ser executadas pelo computador. A etapa de codificação realiza essa conversão." (PRESSMAN, 1995)

a CALDIS foi codificada por meio da linguagem Java, escolha já detalhada no capítulo 2, item 2.2.

Fazendo uso de uma série de regras de testes estabelecidas por Glen Myers (apud PRESSMAN, 1995) que objetivam a aplicação das atividades de testes:

1. A atividade de teste é um processo de executar um programa com a intenção de descobrir um erro.
2. Um bom caso de teste é aquele que tem uma elevada probabilidade de revelar um erro ainda não descoberto.
3. Um teste bem-sucedido é aquele que revela um erro ainda não descoberto.

foram elaboradas simulações de uso da ferramenta CALDIS que serão descritas no capítulo 5.

Os tipos de manutenção empregados na CALDIS serão a manutenção preventiva, a perfectiva e a adaptativa, com vistas a manter a qualidade do mesmo. O primeiro tipo será efetuado mediante a necessidade de se melhorar a confiabilidade ou a manutenibilidade futura do *software* ou de se oferecer uma base mais sólida para melhores instalações; o segundo, a perfectiva, advirá das recomendações de novas capacidades, de modificações em funções existentes e de ampliações gerais recebidas dos usuários; e o terceiro, a adaptativa, implicará na manutenção no *software* a fim de acomodar mudanças em seu ambiente.

A ferramenta de engenharia de *software* utilizada na construção da CALDIS sustenta cada um dos métodos descritos anteriormente à medida que estabelece um sistema de suporte ao desenvolvimento do *software,* combinando "*software, hardware* e um banco de dados de engenharia de *software* (uma estrutura de dados contendo importantes informações sobre análise, projeto, codificação e teste)." (PRESSMAN, 1995)

Os procedimentos de engenharia de *software* para a fundamentação da CALDIS constituíram

> "o elo de ligação para manter juntos os métodos e as ferramentas, possibilitando um desenvolvimento racional e oportuno do *software* em questão; definindo a seqüência em que os métodos eram aplicados, os controles que ajudariam a assegurar a qualidade e a coordenar as mudanças." (PRESSMAN, 1995)

O paradigma de engenharia de *software* escolhido para o desenvolvimento da CALDIS, tendo-se como base a natureza do projeto e da aplicação, os métodos e as

ferramentas que seriam usados, os controles e os produtos que precisariam ser entregues, foi o paradigma do ciclo de vida clássico.

Na figura 4 é ilustrado o paradigma do ciclo de vida clássico da engenharia de *software*, que requer uma abordagem sistemática, seqüencial ao desenvolvimento do *software.* Essa abordagem é iniciada no nível de sistema e avança ao longo da análise, projeto, codificação, teste e manutenção.

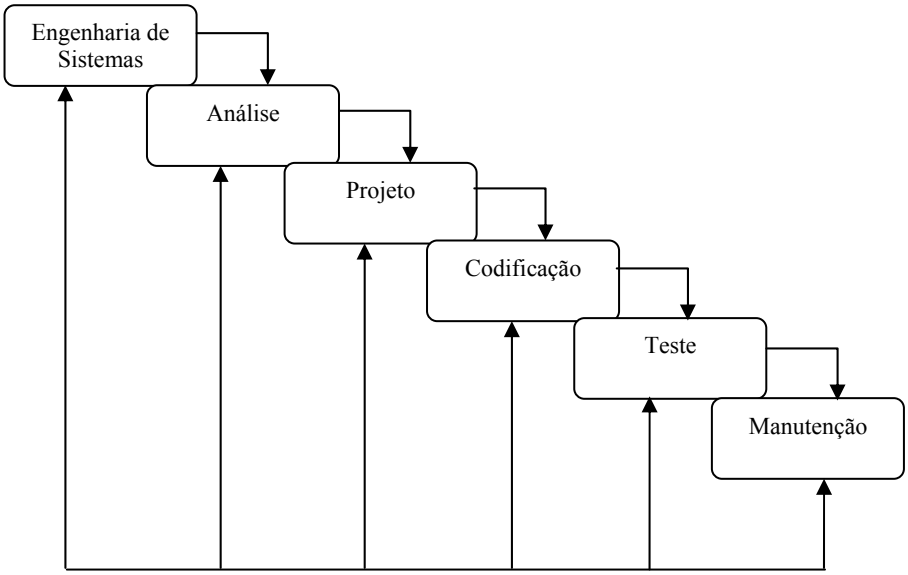

Figura 4 – Paradigma do Ciclo de Vida Clássico

Tão logo os requisitos de *software* foram analisados e especificados, foi iniciado o projeto de *software*, a primeira dentre as três atividades técnicas – projeto, codificação e teste – que são exigidas para se construir e verificar um *software*.

Dentre os requisitos de *software* indicados por Pressman (1995) foram empregados, na construção da CALDIS, os modelos comportamentais, funcionais e de informação.

A importância do projeto de *software* para a construção da CALDIS pode ser estabelecida com uma única palavra: qualidade. “O projeto é o lugar onde a qualidade é

fomentada durante o processo de desenvolvimento. O projeto fornece representações do *software* que podem ser avaliadas quanto à qualidade.” (PRESSMAN, 1995)

Ao longo do processo de projeto em evolução, a qualidade da CALDIS foi avaliada mediante uma série de revisões técnicas formais de projeto. O processo de projeto de engenharia de *software* realizado na CALDIS, por meio da aplicação de princípios fundamentais, metodologia sistemática e revisão cuidadosa, foi responsável pela obtenção de um bom projeto.

A atividade de teste do *software* CALDIS foi considerada um elemento crítico da garantia de qualidade de *software* e representou a última revisão de especificação, projeto e codificação.

Durante todo o processo de desenvolvimento da CALDIS e mesmo atualmente, quando ela já está sendo usada, o processo de manutenção do *software* ocorre porque não é razoável presumir que a atividade de testes de *software* descobrirá todos os erros latentes num sistema de *software*.

O gerenciamento de configuração do *software* foi um conjunto de atividades que foi desenvolvido para administrar as mudanças em todo o ciclo de vida do *software*.

Capítulo 4. Descrição do Trabalho

Para ser realizada a modelagem desse programa foi escolhida a UML, uma linguagem visual utilizada para a modelagem de sistemas computacionais por meio do paradigma de orientação a objetos.

A CALDIS foi implementada por meio da linguagem de programação Java, que se caracteriza por ser gratuita, portável, robusta e segura.

O banco de dados utilizado, MySQL, foi escolhido por ser gratuito, por suportar grandes estruturas de dados e por ser seguro.

A CALDIS está hospedada num servidor *on-line* (http://www.mmurari.com/caldis), portanto, a velocidade da mesma está subordinada à velocidade da conexão da Internet.

Para o uso da CALDIS, não há limitação quanto ao número de professores-usuários, no entanto, quanto maior o número de professores conectados a mesma, mais lento será o tempo de processamento de dados e informações no servidor *on-line*.

4.1. Definição dos Atores

Após a análise da descrição do problema do sistema e o levantamento de requisitos, ficaram determinados quatro atores - administrador, aluno, coordenador e professor - dados na Figura 5.

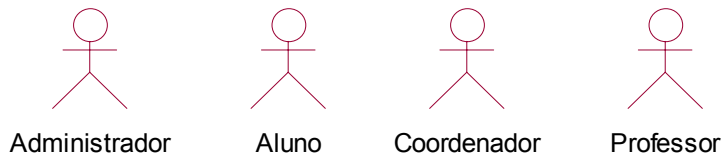

Figura 5 - Definição de Atores.

4.2. Listas de Casos de Uso

Baseados nos requisitos do sistema, foram definidos os casos de uso representados por 12 eventos e por 16 mensagens descritos nas tabelas 1 e 2.

Na tabela 1, são mostrados os eventos que o usuário pode realizar no sistema. Os usuários “administrador”, “aluno”, “coordenador” e “professor”, realizam funções no sistema.

Primeiramente, o “administrador” cadastra, altera e exclui todo tipo de usuário.

O “aluno” possui acesso a todos os textos de apoio a Sistemas Digitais. Em seguida, ele interage com o seu “professor” efetuando perguntas, recebendo e enviando respostas ao mesmo; o “aluno” pode, ainda, escolher os parâmetros a serem executados em um simulador acoplado à CALDIS. O usuário “aluno” cria e salva seus textos em arquivos.

O “coordenador” altera, exclui e aloca os “alunos” para os “professores” correspondentes.

O “professor” visualiza todo o acesso dos seus “alunos”, num processo de interação com os mesmos em que ele faz perguntas, envia e recebe respostas. O “professor” cria e salva seus textos em arquivos e, quando necessário, pode também alterar seus dados cadastrais.

Na tabela 2, são enumeradas as mensagens. Na CALDIS, essas mensagens serão enviadas após uma confirmação de cadastro, uma alteração ou uma exclusão de usuários, após os textos serem salvos, também após uma realocação de alunos e também mensagens de erro quando a senha ou login forem digitados errado.

Tabela 1 - Definição dos Eventos.

Número	Descrição
01	Administrador gerencia usuários
02	Aluno visualiza textos de apoio
03	Aluno interage com professor
04	Aluno escolhe parâmetros
05	Aluno executa simulador
06	Aluno elabora texto
07	Coordenador gerencia alunos

08	Professor visualiza acesso do aluno
09	Professor interage com aluno
10	Professor realiza texto
11	Professor alterar dados cadastrais

Tabela 2 - Definição das Mensagens.

Mensagens	
Msg01/02/03	"Usuário cadastrado com sucesso" / "Usuário alterado com sucesso" / "Usuário excluído com sucesso"
Msg04	"Texto salvo com sucesso"
Msg05	"Aluno realocado com sucesso"
Msg06	"Texto salvo com sucesso"
Msg07	"Dados alterados com sucesso"
Msg08	"Login incorreto"
Msg09	"Senha incorreta"

4.3. Diagramas de Casos de Uso

> "Um diagrama de casos de uso apresenta os casos de uso do sistema. Cada caso de uso será apresentado com os seus diagramas (classes e seqüências) e com a descrição textual do seu comportamento. Um diagrama de casos de uso é um diagrama mais abstrato, informal e flexível da UML. Ele é utilizado para demonstrar o comportamento externo do sistema na visão do usuário, demonstrando as funções que o usuário poderá utilizar." (GUEDES, 2005)

No diagrama de caso de uso, dado na figura 6, são representadas as responsabilidades pertinentes ao ator "administrador". No primeiro caso de uso, chamado "CadastrarUsuário", o "administrador" deverá cadastrar os usuários alunos, professores e coordenador. Para efetuar o cadastro do usuário correspondente, o "administrador" deverá possuir os dados cadastrais do usuário. Quando o cadastro for efetivado, o "administrador" receberá uma mensagem (Msg01) informando que o usuário foi cadastrado com sucesso.

Já no segundo caso de uso, chamado "AlterarUsuário", o "administrador", se necessário, poderá alterar os dados de um usuário já cadastrado, recebendo uma mensagem (Msg02) informando que o(s) referido(s) dado(s) foram alterados com

sucesso.

No último caso de uso deste ator, chamado “ExcluirUsuário”, o “administrador” poderá excluir um usuário. Assim que o usuário for excluído do banco de dados, o “administrador” receberá uma mensagem (Msg03) informando que o usuário foi excluído com sucesso.

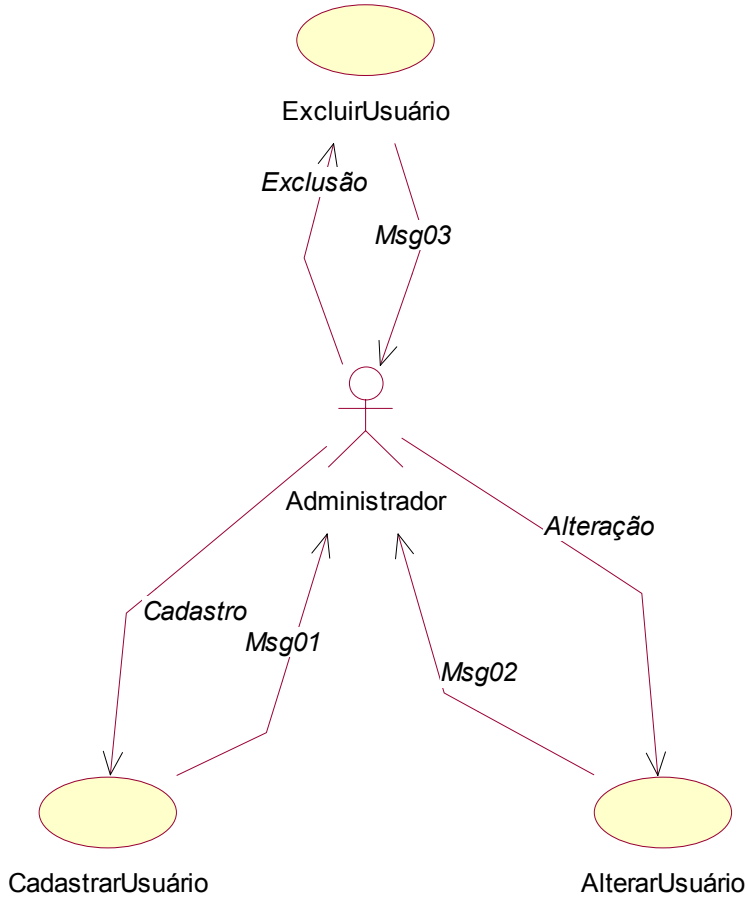

Figura 6 - Diagrama de Caso de Uso do Ator Administrador.

Na Figura 7, é mostrado o diagrama de caso de uso do ator “aluno”, no qual são representadas as funções pertinentes a este ator. No primeiro caso de uso, chamado “VisualizarTextos”, o ator “aluno” poderá solicitar textos de apoio para estudos em Sistemas Digitais. No segundo caso de uso, chamado “InteragirProfessor”, o “aluno” poderá interagir com o seu professor responsável, efetuando e respondendo perguntas pertinentes a matéria estudada. Com a terceira situação do terceiro caso de uso,

chamado “EscolherParâmetros”, o “aluno” poderá escolher os parâmetros a serem executados no simulador acoplado à ferramenta.

No quarto caso de uso, chamado “ExecutarSimulador”, o “aluno” deverá executar os parâmetros escolhidos em um simulador acoplado a ferramenta.

No quinto caso de uso, chamado “RealizarTexto”, o “aluno” poderá redigir um texto contendo observações feitas, por ele, sobre toda sua operação realizada na ferramenta. Estas observações poderão ser salvas em um arquivo texto, podendo ser aberto em uma próxima execução da ferramenta. Quando o arquivo for salvo, o “aluno” receberá uma mensagem (Msg04) informando que o texto foi salvo com sucesso.

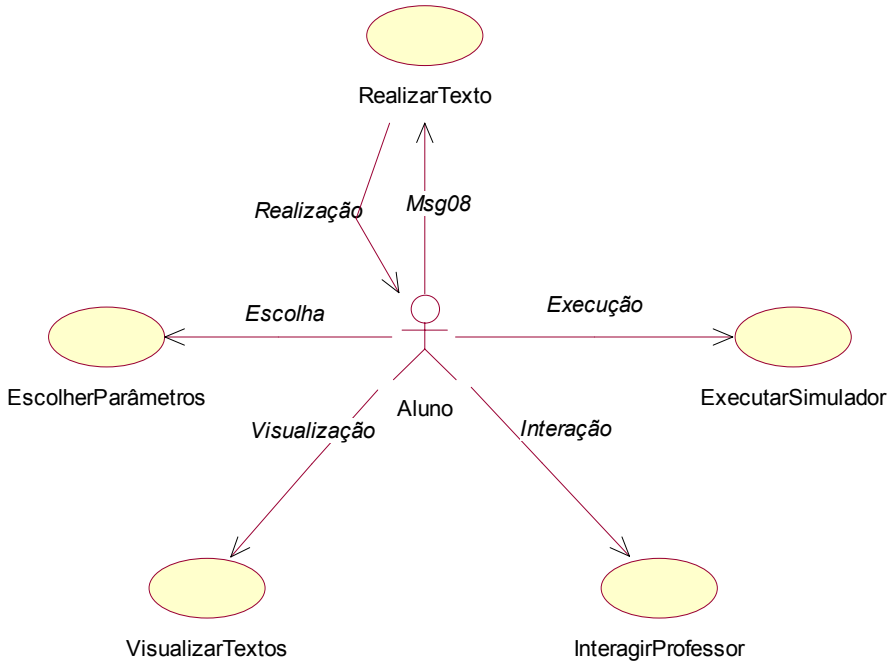

Figura 7 - Diagrama de Caso de Uso do Ator Aluno.

No diagrama de caso de uso, dado na figura 8, são representadas as responsabilidades pertencentes ao ator “coordenador”. No caso de uso, chamado “RealocarAluno”, o “coordenador” deverá realocar os alunos do professor que foi desvinculado do sistema e realocá-lo para outro professor. Quando a realocação for

efetuada, o “coordenador” receberá uma mensagem (Msg05) informando que o aluno foi realocado com sucesso.

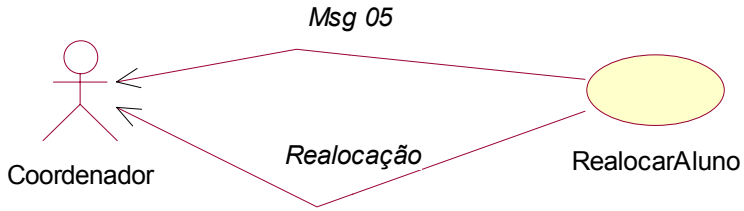

Figura 8 - Diagrama de Caso de Uso do Ator Coordenador.

No diagrama de caso de uso, dado na figura 9, é exibido o diagrama de caso de uso do ator “professor”, no qual são representadas as funções atribuídas a este ator. No primeiro caso de uso, chamado “VisualizarAcesso”, o “professor” poderá visualizar o acesso realizado, por seus alunos, à ferramenta. No segundo caso de uso, chamado “InteragirAluno”, o “professor” poderá interagir com seus alunos, efetuando e respondendo perguntas pertinentes a matéria abordada.

Com a situação do terceiro caso de uso, chamado “RealizarTexto”, o “professor” poderá redigir um texto contendo observações, por ele efetuadas, sobre toda a operação realizada, por ele, na ferramenta; estas observações poderão ser salvas em um arquivo texto, que poderá ser aberto em uma próxima execução da ferramenta. Quando o arquivo for salvo, o “professor” receberá uma mensagem (Msg06) informando que o texto foi salvo com sucesso.

Já no último caso de uso chamado “AlterarCadastro”, o “professor” poderá, quando necessário, alterar seus dados cadastrais. Após a alteração do seu cadastro, o “professor” receberá uma mensagem (Msg07) afirmando que os seus dados cadastrais foram alterados com sucesso.

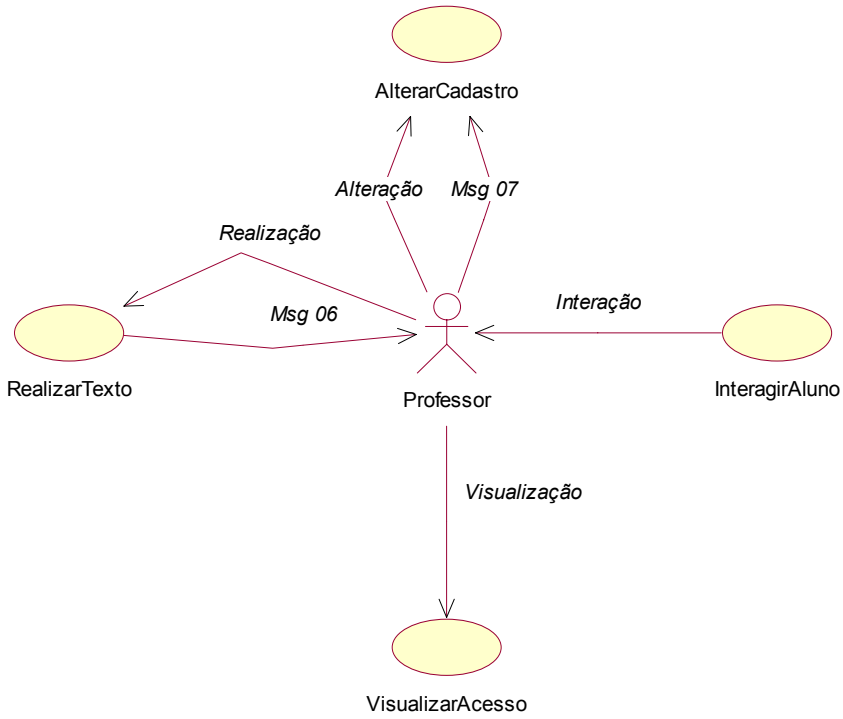

Figura 9 - Diagrama de Caso de Uso do Ator Professor.

4.4. Diagrama de Classes

> "O diagrama de classes, dado na figura 10, é o diagrama mais importante na UML e serve como base para outros diagramas. Este diagrama define a estrutura das classes utilizadas pelo sistema, determinando os atributos e métodos possuídos por cada classe, além de estabelecer como as classes se relacionam e trocam informações entre si." (GUEDES, 2005)

> "Com os diagramas de classes é possível visualizar um conjunto de classes, *interfaces* e colaborações, bem como os relacionamentos entre as classes. Graficamente, um diagrama de classes é uma coleção de vértices e arcos. Ele é usado para fazer a modelagem da visão estática do projeto de um sistema." (BOOCH; JACOBSON; RUMBAUGH, 2000).

Os diagramas de classes são importantes não só para a visualização, a especificação e a documentação de modelos estruturais, mas também para a construção de sistemas executáveis. Eles podem conter pacotes ou subsistemas, utilizados para

agrupar elementos do modelo deles em um conjunto maior e para visualizar o tipo de uma instância.

Ao ser criado um diagrama de classes, uma parte dos itens e relacionamentos que compõem a visão de projeto do sistema é modelada, assim é focada apenas uma colaboração por vez.

Um diagrama de classes bem estruturado é composto de um foco para comunicar um único aspecto da visão estática do projeto de sistema, contém somente elementos essenciais à compreensão desse aspecto, fornece detalhes consistentes com o respectivo nível de abstração, exibi somente os adornos essenciais à compreensão e não é tão minimalista que prejudique a informação do leitor sobre a semântica importante.

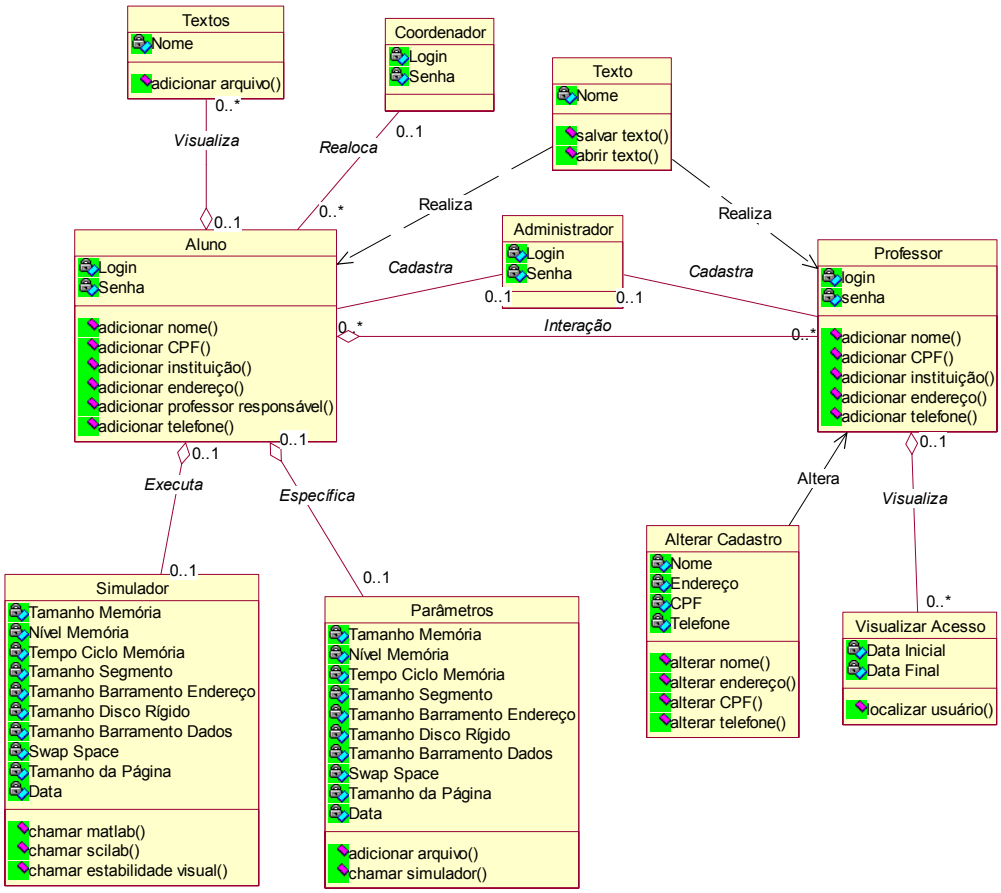

Figura 10 - Diagrama de Classes.

Existem dez classes que compõem nosso ambiente: “textos”, “texto”, “coordenador”, “aluno”, “administrador”, “professor”, “simulador”, “parâmetros”, “alterar cadastro”, “visualizar acesso”.

A classe “textos” é agregada somente à classe “aluno”, pois a mesma é responsável pela visualização de textos de apoio pelos alunos. Neste sentido, é permitida ao usuário aluno a visualização de vários textos simultaneamente.

Na classe “texto”, a qual possui uma dependência com a classe “professor” e com a classe “aluno”, uma vez que apresenta apenas a função de elaboração de textos pelos professores e alunos, é permitida a criação de textos não só por aqueles usuários, mas também por estes.

Já a classe “coordenador”, que se relaciona somente com a classe “aluno”, relacionamento condicionado pela responsabilidade hierárquica existente entre os atores “coordenador” e “professor”, permite que somente o “coordenador” aloque alunos para determinado “professor”.

A classe “aluno”, que recebe uma agregação das classes “textos”, “simulador”, “parâmetros” e “professor”, possui dependência em relação à classe “texto” e, ainda, possui um relacionamento com a classe “administrador” dependendo totalmente desta para a efetuação de cadastro.

Quanto à classe “administrador”, é verificado que a mesma está relacionada com a classe “aluno” e com a classe “professor”, permitindo assim que o “administrador” do sistema cadastre vários professores e alunos.

A classe “professor” recebe uma agregação da classe “visualizar acesso” e das classes “alterarcadastro” e “texto” no tocante as atividades de realização de textos e de alteração de cadastros.

Na classe “simulador”, é apresentada apenas uma agregação com a classe

“aluno”, ou seja, o usuário aluno executa o simulador.

Já na classe “parâmetros”, é realizada uma agregação com a classe “aluno”, pois o aluno, por sua vez, define os parâmetros a serem executados no simulador.

Na classe “alterar cadastro”, é apresentada apenas uma dependência em relação às classes “professor” e “aluno”, pois tanto os professores quanto os alunos poderão alterar seus dados cadastrais.

Na classe “visualizar acesso”, é mostrada somente uma agregação com a classe “professor”, agregação que permite ao professor visualizar o acesso dos alunos dele.

Nesta seção foi descrita a estrutura das classes que formam a ferramenta CALDIS. Na próxima seção, será estudado o diagrama de seqüência, o qual nos mostrará a ordem temporal em que os métodos das classes são chamados.

4.5. Diagramas de Seqüência

> “O diagrama de seqüência mostra uma interação que dá ênfase à ordenação temporal das mensagens e à organização estrutural dos objetos que enviam e recebem mensagens. Ele é utilizado para fazer a modelagem dos aspectos dinâmicos do sistema. Isso envolve a modelagem de instâncias concretas ou prototípicas de classes, *interfaces*, componentes e nós, juntamente com as mensagens que são trocadas, tudo isso no contexto de um cenário que ilustra um comportamento.” (BOOCH; JACOBSON; RUMBAUGH, 2000).

No diagrama de seqüência, é permitido visualizar, especificar, construir e documentar a dinâmica de uma determinada sociedade de objetos e modelar um determinado fluxo de controle de um caso de uso.

Não é importante somente para a modelagem de aspectos dinâmicos do sistema, mas também para a construção de sistemas executáveis.

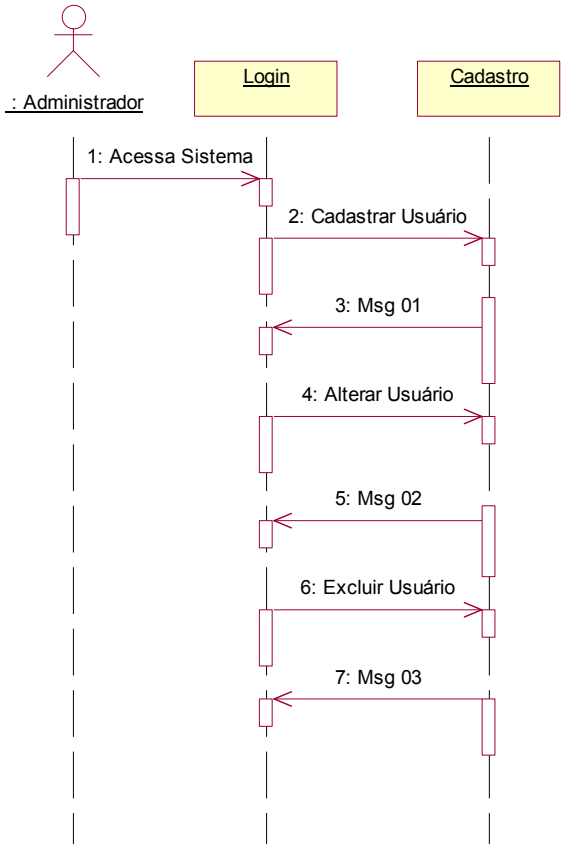

Figura 11 - Diagrama de Seqüência do Ator Administrador.

Como é observado no diagrama de seqüência, dado na figura 11, para o cadastro dos usuários à ferramenta, faz-se necessário que o "administrador" do sistema cadastre usuários, aquele também, por sua vez, poderá alterar e excluir estes usuários quando achar necessário. Caso ocorra algum erro durante o cadastro, a alteração ou a exclusão dos usuários, o próprio sistema retorna uma mensagem informando que a ação não pôde ser completada.

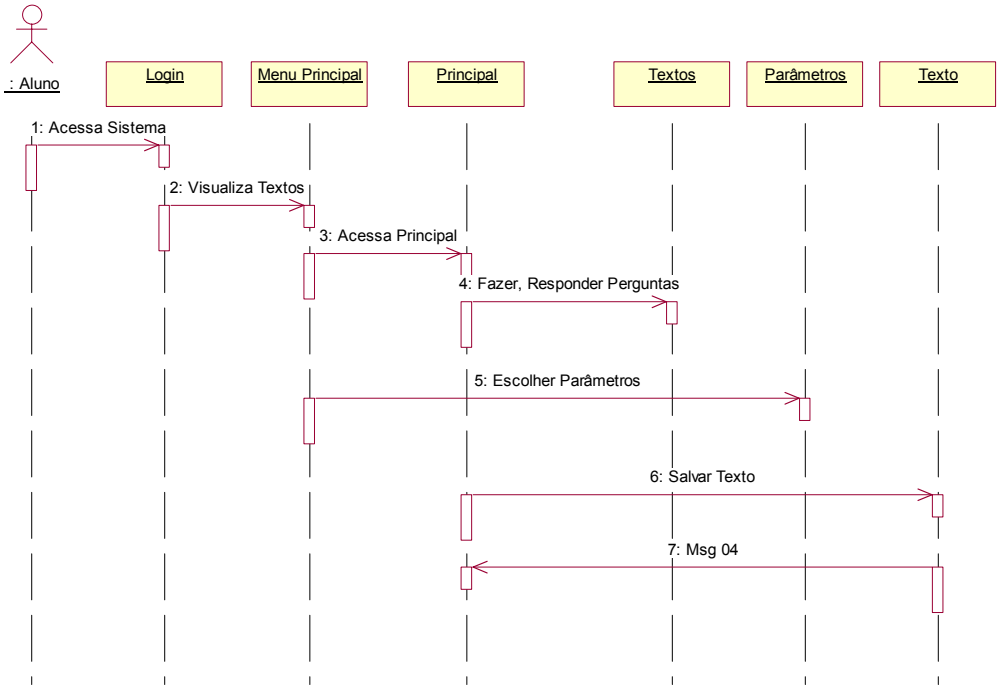

Figura 12 - Diagrama de Seqüência do Ator Aluno.

Já no diagrama de seqüência, dado figura na figura 12, para que o "aluno" acesse a ferramenta, ele tem que efetuar o seu *login* e senha, se o *login* e a senha não estiverem corretos, o sistema enviará uma mensagem informando o erro, caso contrário, o "aluno" terá acesso livre à CALDIS; a partir deste ponto o "aluno" terá acesso ao menu principal, onde poderá escolher textos de apoio em Sistemas Digitais para auxiliá-lo em seus estudos.

Logo após a visualização dos textos de apoio, o "aluno" terá acesso à tela principal, a qual oferecerá vários recursos para o desenvolvimento do trabalho desse ator e para a resolução de suas dúvidas. Uma dessas opções é a tela de perguntas e respostas, onde o "aluno" poderá efetuar e responder perguntas ao "professor" responsável por ele.

A ferramenta possibilita ao "aluno" criar e salvar suas observações em um arquivo texto. Depois de salvo o arquivo, será enviada ao aluno uma mensagem dizendo

que o arquivo foi salvo com sucesso.

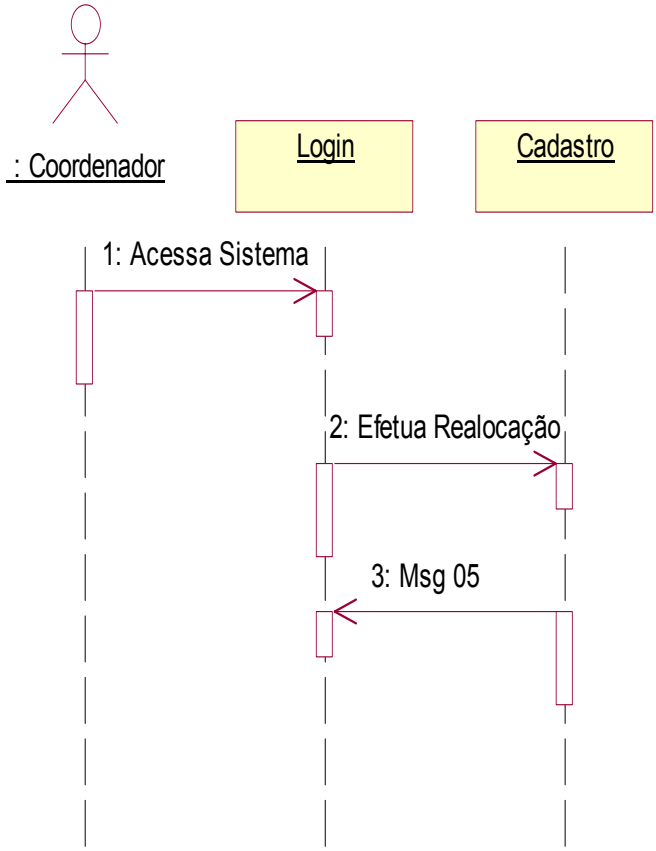

Figura 13 - Diagrama de Seqüência do Ator Coordenador.

Como pode ser observado no diagrama de seqüência, dado na figura 13, para a realocação de usuários alunos à ferramenta, faz-se necessário que o "coordenador" do sistema efetue essas realocações. Após a realocação, será enviada uma mensagem dizendo que o aluno foi realocado com sucesso.

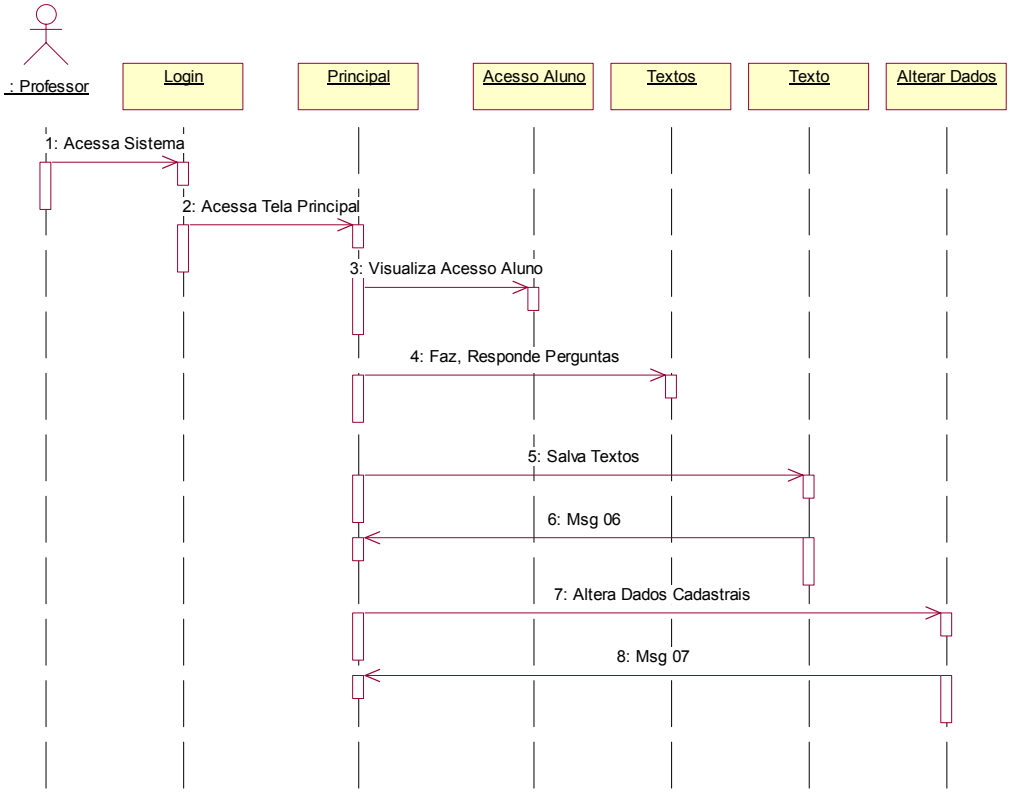

Figura 14 - Diagrama de Seqüência do Ator Professor.

No diagrama de seqüência, dado na figura 14, o ator professor acessa a ferramenta, efetuando *login* e senha. Se o *login* e a senha não estiverem corretos, o sistema enviará uma mensagem informando o erro, caso contrário, ele terá acesso livre ao ambiente. A partir deste ponto, o "professor" acessará o menu principal, onde poderá visualizar o acesso à ferramenta efetuado por seus alunos.

O "professor" terá, ainda, acesso a tela de perguntas e respostas, onde poderá efetuar e responder perguntas para seus alunos. O "professor" poderá ainda criar e salvar suas observações em um arquivo texto. Depois de salvo o arquivo, será enviada a ele uma mensagem dizendo que o arquivo foi salvo.

E por fim, o "professor" poderá alterar seus dados cadastrais, se necessário for; e após a sua alteração, será enviada a ele uma mensagem avisando que seus dados cadastrais foram alterados com sucesso.

Capítulo 5. Resultados Obtidos

Considerando que

> "o desenvolvimento de sistemas de *software* envolve uma série de atividades de produção em que as oportunidades de injeção de falhas humanas são enormes. Erros podem começar a acontecer logo no começo do processo, onde os objetivos... podem estar errônea ou imperfeitamente especificados, além de erros que venham a ocorrer em fases de projeto e desenvolvimento posteriores... Por causa da incapacidade que os seres humanos têm de executar e comunicar com perfeição, o desenvolvimento de *software* é acompanhado por uma atividade de garantia de qualidade." (DEUTSCH, 1979 apud PRESSMAN, 1995)

foram elaboradas e aplicadas atividades de teste para os atores "professor" e "aluno", com o intuito de realizar uma análise acerca da confiabilidade da CALDIS.

Os testes aqui descritos foram realizados em três Instituições de Ensino Superior; uma localizada em Fernandópolis-SP – FEF - Fundação Educacional de Fernandópolis, outra em Jales-SP – UNIJALES - Centro Universitário de Jales e outra em Frutal – SP – Universidade do Estado de Minas Gerais, nos meses de fevereiro e março, setembro e outubro de 2007.

A escolha dessas instituições deveu-se ao fato das mesmas apresentarem-se como instituições mais acessíveis ao cumprimento das exigências de instalação desse sistema, como, por exemplo, a desativação do *firewall* – um filtro de informações que entram e saem da *extranet*, exigência determinada pelo bloqueio criado à gravação dos dados em um banco de dados externo à *extranet* o que impediria o armazenamento de dados e informações intrínsecos à CALDIS.

A aplicação dos testes foi realizada nos laboratórios de informática das instituições, por meio de questionários direcionados para professores e alunos, cujos

teores e resultados gráficos poderão ser melhor observados nos Anexos I e II ao final desse trabalho.

Da aplicação dos testes e do preenchimento das pranchas de testes da CALDIS pelos professores, foi verificada uma grande aceitação desse sistema enquanto:

> "... uma possibilidade de interação entre o conhecimento e a assimilação em Sistemas Digitais e, ainda, entre os professores e os alunos envolvidos nesse processo."
> "... um recurso pedagógico facilitador e dinamizador do processo de ensino-aprendizagem de Engenharia."
> "... recurso pedagógico dinâmico, prático e confiável."

Na questão 1, foi observado que a CALDIS apresenta uma interface visual agradável ao seu manuseio; na questão 2, a CALDIS, enquanto uma ferramenta de apoio à aprendizagem, apresentou-se funcional; na questão 3, foi verificado que os recursos tecnológicos empregados na CALDIS, tais como a linguagem de programação Java, um banco de dados MySql, integração com a *web* e o funcionamento em multiplataformas, ajudam na assimilação dos assuntos por ela abordados; na questão 4, a CALDIS mostrou ser um sistema confiável quanto à integridade dos dados; na questão 5, a aplicação da CALDIS para o ensino foi considerada boa; na questão 6, a forma como a ferramenta é apresentada e as funcionalidades por ela oferecidas são consideradas satisfatórias a um enriquecimento pedagógico do trabalho docente; na questão 7, a CALDIS foi aceita enquanto um recurso pedagógico dinâmico, prático e confiável para o processo ensino-aprendizagem. A média atribuída à CALDIS pelos professores foi 8,5.

A mesma aceitação foi averiguada pelos alunos mediante a aplicação dos testes e o preenchimento das pranchas de testes da CALDIS:

"... um excelente sistema, pois proporciona ajuda, é confiável, prático e ótimo para o processo ensino-aprendizagem."

"... ferramenta fácil de mexer e com bastante recursos disciplinares..."

"... um sistema muito seguro e simples para utilizar."

"... sistema interessante, com a evolução correta terá muito a acrescentar."

"... o sistema será bem útil para os alunos."

"...é necessário um melhor planejamento na parte visual da ferramenta, assim como a implementação de validações e criptografias para assegurar a segurança dos dados. Também é necessário implementar novos critérios para verificar se o aluno fez ou não as atividades propostas pelo professor!"

"...Acho que a CALDIS é excelente para o ensino a distancia, penso que uma forma de melhorar esse sistema seria criar uma maior interatividade para o usuário, além da sua funcionalidade ele também poderia trazer informações da unidade de ensino, ou também trazer formas de aumentar a competitividade em relação ao grau de entendimento dos alunos".

"...Excelente, não deixa a desejar".

"...Melhorar interfaces. É uma ferramenta computacional muito boa".

"...Depois de alguns aperfeiçoamentos, ela será uma ótima ferramenta de apoio aos professores".

"...É uma ferramenta interativa, mas na parte de segurança ela me parece não ser confiável".

"...A CALDIS mereceria nota 10, se tivesse uma interface mais agradável e intuitiva, pois a funcionalidade aparenta ser muito boa".

"...Tem uma ótima interação entre aluno e professor".

"...Facilita o entendimento do aplicativo, das atividades".

"...Ferramenta inovadora que propicia inúmeras funcionalidades no processo de aprendizagem de sistema digitais. Talvez poderia deixar o *layout* mais atrativo".

"...Melhorar o *layout* para auxiliar na utilização da ferramenta".

"...A interface pode ser melhorada fazendo uso de ícones que representem as informações acessadas/utilizadas pelos usuários, poderia haver mais mensagens de status. O sistema é importante, pois centraliza as atividades práticas e aproxima professores e aluno, muito boa a idéia".

"...Sobre a questão dos simuladores via *web*, o sistema realmente ficaria a desejar nesse sentido, fora isso está ótimo".

"...A CALDIS, além de proporcionar ajuda ao usuário, é confiável e dinâmica, ou seja, é um ótimo sistema".

> "...Ótima ferramenta para inclusão digital e interação entre professores e alunos".
>
> "...É muito pesada para *web*. Na hora de excluir o professor e adquirir o professor deveria atribuir os alunos de uma só vez".
>
> "...Por ser um sistema *web* ele é muito pesado".
>
> "...Melhorar estética, navegabilidade e validações".
>
> "...A única observação seria quanto à interface, que é muito simples. Mas como já explicado que não houve preocupação com essa parte, só posso dizer que é boa, funcional e confiável".
>
> "...A interface visual da CALDIS não é agradável para o manuseio, mas o sistema em si está muito completo. Aprendi muito e obtive muitas informações importantes.
>
> "...Verificar as validações dos dados, a integridade dos dados no banco de dados e melhorar a interface".
>
> "...Um sistema muito interessante, que pode ser melhorado, mas visualmente, traz uma boa aparência ao usuário".
>
> "...Em minha opinião, acho que o sistema pode diminuir um pouco a distância entre o aluno e o professor e, ainda, dar uma oportunidade ao aluno de tirar dúvidas que não tem coragem de perguntar ao professor em sala".
>
> "...Eu achei o sistema muito prático e interessante e poderia auxiliar muito no aprendizado. Mais achei dificuldade na interface e na colocação de botões e funções do sistema".
>
> "...A senha não está criptografada, alguns botões poderiam ser mais específicos".

Na questão 1, a maioria dos alunos considerou razoável a ajuda que a CALDIS propiciou na compreensão dos conteúdos ministrados em sala de aula; na questão 2, a maioria dos alunos qualificou como muito agradável o manuseio da interface visual da CALDIS; na questão 3, foi considerado que os recursos tecnológicos empregados na CALDIS, tais como a linguagem de programação Java, um banco de dados MySql, integração com a *web* e o funcionamento em multiplataformas, ajudam na assimilação dos assuntos por ela abordados; na questão 4, mais da metade dos alunos definiu como razoável o nível de confiabilidade apresentado pela CALDIS; na questão 5, a aplicação

da CALDIS para a aprendizagem foi tida como adequada; na questão 6, a maioria dos alunos classificou a CALDIS como um sistema bom quanto ao oferecimento de conhecimentos complementares aos já adquiridos; na questão 7, mais da metade dos alunos classificou a CALDIS como um recurso pedagógico dinâmico, prático e confiável para o processo ensino-aprendizagem. A média atribuída à CALDIS pelos alunos foi 8,4.

Atribuímos a análise dos resultados aqui apresentada à ajuda dos professores de estatística, Profa. Ms. Elen Cristina Mazucchi da Fundação Educacional de Fernandópolis e Prof. Ms. Clinton André Merlo do Centro Universitário de Jales, ajuda despendida não só na quantificação dos resultados, mas também na análise e alteração das pranchas iniciais dos testes.

Considerações Finais

Ao estudar o processo ensino-aprendizagem nos seus aspectos pedagógicos, sociais, políticos e econômicos, as considerações apresentadas neste trabalho reconhecem o aluno enquanto um sujeito ativo desse processo; um sujeito fruto de uma realidade concreta e dotado de experiências significativas que o tornam um ser humano único com capacidades e ritmos próprios de aprendizagem.

Nessa perspectiva, o estilo de aprendizagem de David Kolb aqui descrito foi escolhido por respeitar o ritmo de desenvolvimento de cada aluno que experimenta (experiência concreta), observa (observação reflexiva), cria (conceitualização abstrata) e toma decisões (experimentação ativa) - ciclo quadrifásico do *modelo de aprendizagem vivencial* desse estudioso que se aplica à utilização e à compreensão da CALDIS - num processo ativo de elaboração e assimilação do conhecimento.

Como a ferramenta CALDIS foi desenvolvida na linguagem Java, ela pode ser executada em qualquer plataforma *hardware/software* que execute a JVM (*Java Virtual Machine*), característica relevante e indispensável se considerada a possível demanda pela execução da CALDIS nas mais variadas plataformas. Aliadas a essa característica, a confiabilidade e a segurança oferecidas pela Java proporcionam à CALDIS por meio, respectivamente, da verificação de tipos de dados em tempo de compilação e da autenticação do usuário, a proteção das informações nele disponibilizadas.

Durante a aplicação dos testes, foi verificado que os *applets* do Java apresentaram problemas para serem executados a partir de redes com *firewalls*, pois, estas interpretam os *applets* como ataques em potencial. Sendo assim, fez-se necessário solicitar ao administrador da rede, das instituições mencionadas, a liberação dos *applets* pelo *firewall.* Quanto a essa condição, é vislumbrado, que, em um trabalho futuro, o

desenvolvimento da CALDIS dê-se em JSP, visto que a execução dessa linguagem é própria do servidor e não do cliente.

Por meio da aplicação das atividades de testes e, conseqüentemente, da análise dos resultados obtidos, foi verificado que a CALDIS, enquanto uma ferramenta de apoio ao ensino de Sistemas Digitais, é tida como um recurso pedagógico centrado no aprendizado interativo, dinâmico e contextualizado, no qual professores e alunos quantificam e qualificam o objeto de estudo em Engenharia, considerando, ainda, mediante reestruturações que atendam às diferentes exigências teórico-metodológicas, a viabilidade do emprego desta ferramenta em outras áreas do ensino, não se restringindo, portanto, a Sistemas Digitais nem a Ciências Exatas.

Referências Bibliográficas

ALDE, L. **Educação a distância:** em que ponto estamos? [s.l.: s.n], 2003.
Disponível em: http://www.elearningbrasil.com.br/home/artigos; Acesso em: 10/03/2007, 15h30min.

ARAÚJO, T. L. & MALTEZ, L. G. M. **Educação a distância:** retrospectiva histórica. 7 ed. [s.l.]: Nexus, 2003.
Disponível em: http://www.elearningbrasil.com.br/home/artigos; Acesso em: 10/03/2007, 15h00min.

BAUER, F. ***Software engineering:*** a report on a conference sponsored by the nato science committee. [s.l.: s.n], 1969. p. 150

BIGGE, M. L. **Teorias da aprendizagem para professores.** São Paulo: Pedagógica e Universitária Ltda., 1977. p. 80

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. **UML – Guia do Usuário.** Rio de Janeiro: Campus, 2000. p. 120

CORNELL, G. & STRAIN, T. **Delphi Segredos e Soluções.** São Paulo: Makron Books. 1995. p. 88

DUNN, R.; DUNN, K. ***Dispelling outmoded beliefs about student learning***. educational leadership. São Paulo, 1987. p. 75

FURGERI, S. **Java 2:** ensino didático: desenvolvendo e implementando aplicações. São Paulo: Érica, 2002. p. 112

GUEDES, G. T. A. **UML 2:** guia de consulta rápida. São Paulo: Novatec Editora Ltda, 2005. p. 154

GUIMARÃES, C. B. **Três milhões aprendem no Brasil, mesmo longe do professor.** São Paulo: Gazeta Mercantil, 2004.
Disponível em: http://www.elearningbrasil.com.br/home/artigos; Acesso: 10/03/2007, 15h18min.

KERNIGHAN, B. W.; RITCHIE, D. M. **C:** a linguagem de programação padrão **ANSI.** Rio de Janeiro: Campus, 1989. p. 110

KOLB, D. A. ***Experiential learning theory bibliography* 1971-2001**. Boston Ma.: McBer, [s.d.].

_____. **Psicologia organizacional –** uma abordagem vivencial. São Paulo: Atlas, 1978. p. 92

_____. ***Experiential learning:*** experience as the source of learning and development. New Jersey: Prentice Hall, 1984. p. 73

_____. ***Learning style inventary technical manual***. Boston: McBer and Company. (Revisado em 1989). p. 82

MEEK, B.; HEATH, P. ***Guide to good programming*.** Halstead: Press (Wiley), 1980. p. 156

PRESSMAN. Roger S. **Engenharia de *software*.** São Paulo: Makron Books, 1995. p. 330

RIBEIRO-JUNIOR, A. **Diferenciais da linguagem java.** Campinas: IPEP, 2003. p. 93

SANTOS, A. A. A.; BARIANI D. C. I.; CERQUEIRA S. C. T. **Estilos cognitivos e estilos de aprendizagem.** Campinas: Unicamp, 1997. p. 103

SCHILDT, H. **C, Completo e Total.** São Paulo: Makron Books, 1996. p. 255

SEBESTA, W. R. **Conceitos de linguagens de programação.** 5ª ed. Porto Alegre: Editora Bookman, 2003. p. 143

VASCONCELLOS, C. S. **Planejamento:** projeto de ensino-aprendizagem e projeto político-pedagógico. 13 ed. São Paulo: Libertad, 1995. p. 135

Anexos

Anexo I:

TESTES – CALDIS (PROFESSOR)

1. A CALDIS apresenta uma interface visual agradável ao seu manuseio?
   ( ) Boa ( ) Regular ( ) Ruim

2. A CALDIS, enquanto uma ferramenta de apoio à aprendizagem, apresentou-se funcional?
   ( ) Boa ( ) Regular ( ) Ruim

3. Os recursos tecnológicos empregados na CALDIS, tais como a linguagem de programação Java, um banco de dados MySql, integração com a *web* e o funcionamento em multiplataformas, ajudam na assimilação dos assuntos por ela abordados?
   ( ) Boa ( ) Regular ( ) Ruim

4. A CALDIS mostrou-lhe ser um sistema confiável quanto a integridade dos dados?
   ( ) Boa ( ) Regular ( ) Ruim

5. Na sua opinião, a aplicação da CALDIS para o ensino é:
   ( ) Boa ( ) Regular ( ) Ruim

6. A forma como a ferramenta é apresentada e as funcionalidades por ela oferecidas são satisfatórias a um enriquecimento pedagógico de seu trabalho?
   ( ) Boa ( ) Regular ( ) Ruim

7. Você acredita que ferramentas computacionais (como a CALDIS) comportam-se como um recurso pedagógico dinâmico, prático e confiável para o processo ensino-aprendizagem?
   ( ) Boa ( ) Regular ( ) Ruim

8. Atribua uma nota de 0 a 10 para a CALDIS.
Nota: ___.

9. Deixe aqui as suas observações sobre a CALDIS.

Resultados dos testes da CALDIS aplicados aos Professores

Tabela 3 – Questão 1 do Ator Professor.

Questão 1		
	Freqüência	%
Boa	2	33,00
Regular	4	67,00
Ruim	0	0

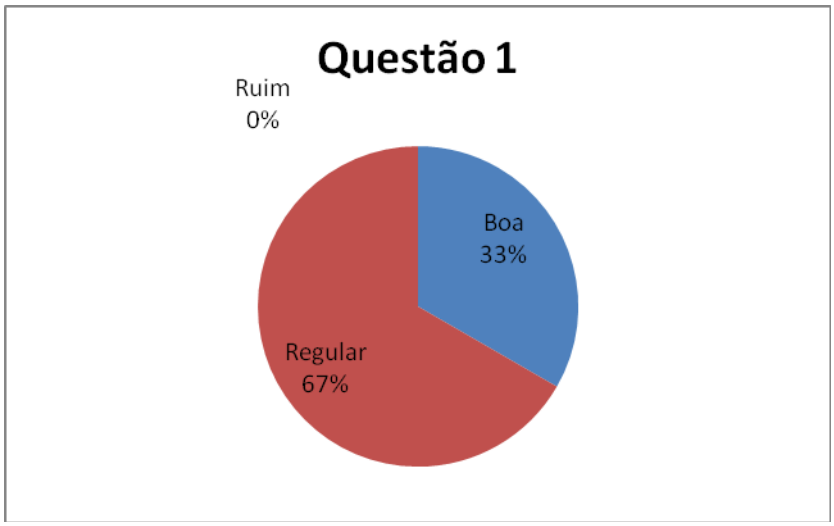

Figura 15 – Gráfico da Questão 1 do Ator Professor.

Tabela 4 - Questão 2 do Ator Professor.

Questão 2		
	Freqüência	%
Boa	4	67,00
Regular	2	33,00
Ruim	0	0

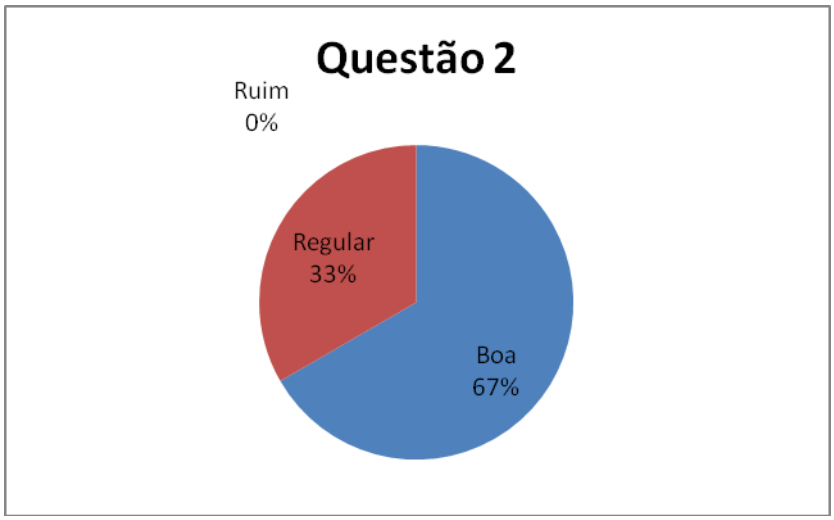

Figura 16 – Gráfico da Questão 2 do Ator Professor.

Tabela 5 - Questão 3 do Ator Professor.

Questão 3		
	Freqüência	%
Boa	3	50,00
Regular	3	50,00
Ruim	0	0

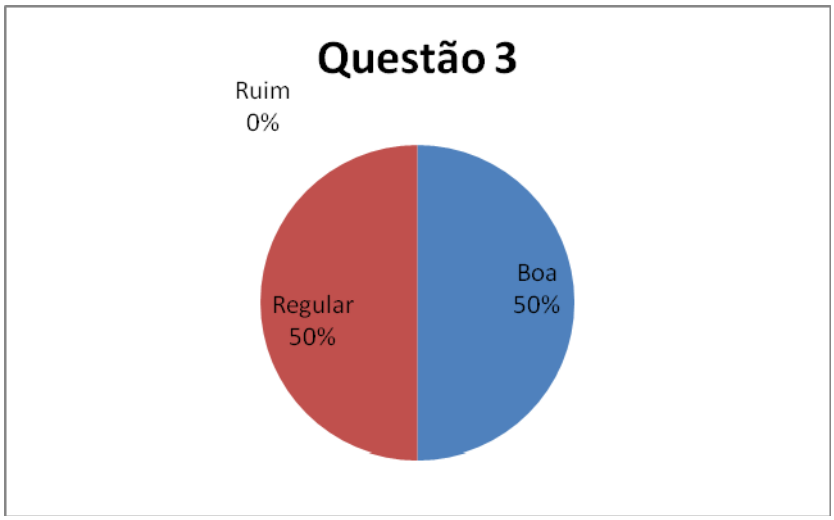

Figura 17 – Gráfico da Questão 3 do Ator Professor.

Tabela 6 - Questão 4 do Ator Professor.

Questão 4		
	Freqüência	%
Boa	1	17,00
Regular	5	83,00
Ruim	0	0

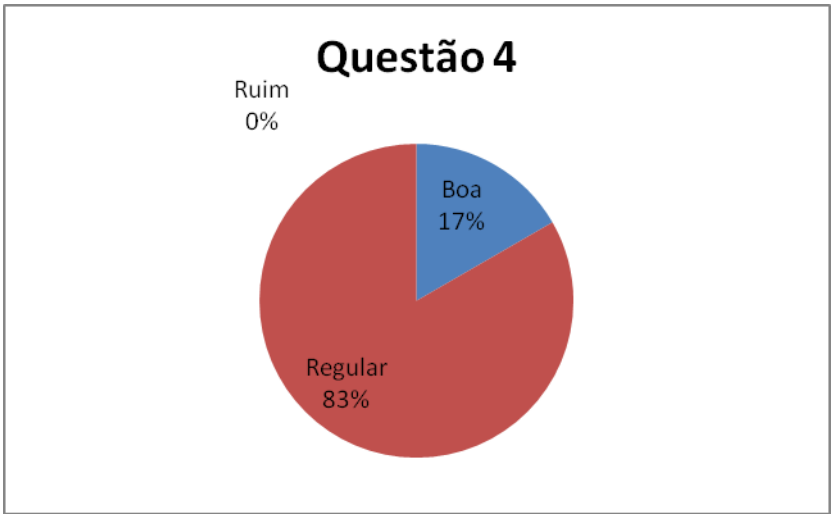

Figura 18 – Gráfico da Questão 4 do Ator Professor.

Tabela 7 - Questão 5 do Ator Professor.

Questão 5		
	Freqüência	%
Boa	4	67,00
Regular	2	33,00
Ruim	0	0

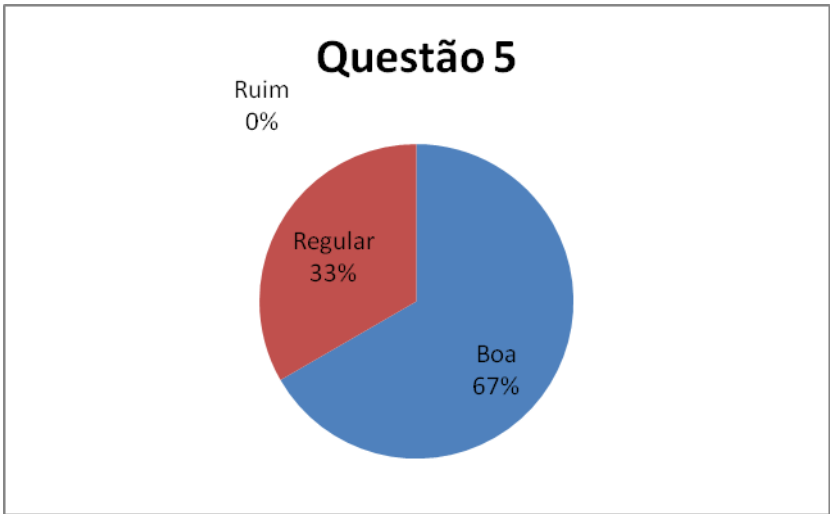

Figura 19 – Gráfico da Questão 5 do Ator Professor.

Tabela 8 - Questão 6 do Ator Professor.

Questão 6		
	Freqüência	%
Boa	4	67,00
Regular	2	33,00
Ruim	0	0

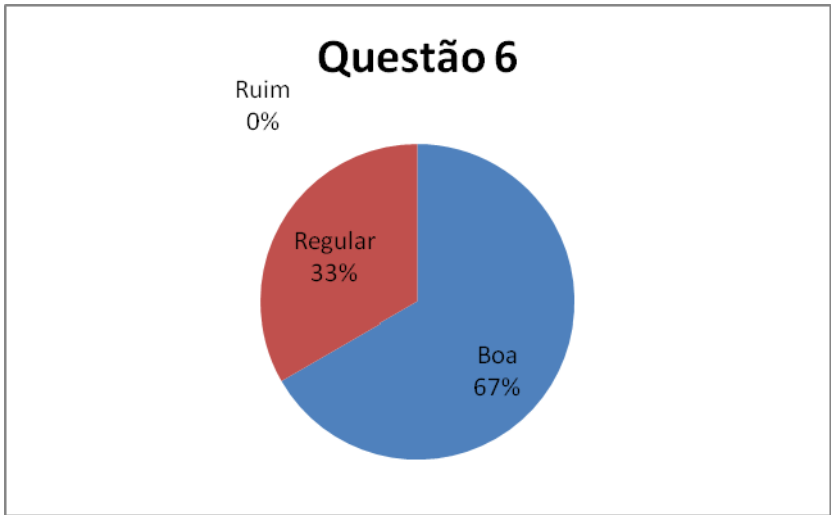

Figura 20 – Gráfico da Questão 6 do Ator Professor.

Tabela 9 – Questão 7 do Ator Professor.

Questão 7		
	Freqüência	%
Boa	4	67,00
Regular	2	33,00
Ruim	0	0

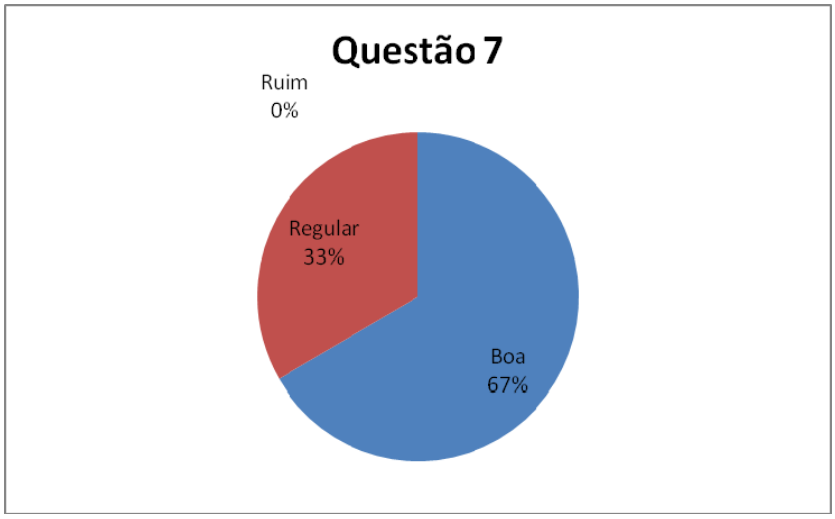

Figura 21 – Gráfico da Questão 7 do Ator Professor.

Tabela 10 - Média do Ator Professor.

MÉDIA	
Professor 1	9,0
Professor 2	9,0
Professor 3	9,5
Professor 4	8,0
Professor 5	8,5
Professor 6	7,5
Média	8,583333
Média Arredondada	8,5

Anexo II:

TESTES – CALDIS (ALUNO)

1. Considerando os conteúdos ministrados nas disciplinas que você cursou na área de sistemas digitais, a CALDIS lhe propiciou alguma ajuda na compreensão dos mesmos?
   ( ) Boa ( ) Regular ( ) Ruim

2. A CALDIS apresenta uma interface visual agradável ao seu manuseio?
   ( ) Boa ( ) Regular ( ) Ruim

3. Os recursos tecnológicos empregados na CALDIS, tais como a linguagem de programação Java, um banco de dados MySql, integração com a *web* e o funcionamento em multiplataformas, ajudam na assimilação dos assuntos por ela abordados?
   ( ) Boa ( ) Regular ( ) Ruim

4. A CALDIS lhe mostrou ser um sistema confiável quanto à integridade dos dados?
   ( ) Boa ( ) Regular ( ) Ruim

5. Na sua opinião, o nível de aplicação de aprendizagem incorporada a CALDIS é:
   ( ) Boa ( ) Regular ( ) Ruim

6. Você acha que as informações contidas na CALDIS complementaram seus conhecimentos acerca de determinados assuntos?
   ( ) Boa ( ) Regular ( ) Ruim

7. As ferramentas computacionais (como a CALDIS), na sua opinião, comportam-se como um recurso pedagógico dinâmico, prático e confiável para o processo ensino-aprendizagem?
   ( ) Boa ( ) Regular ( ) Ruim

8. Atribua uma nota de 0 a 10 para a CALDIS.
Nota: ___.

9. Deixe aqui as suas observações sobre a CALDIS.

Resultados dos testes da CALDIS aplicados aos Alunos

Tabela 11 - Questão 1 do Ator Aluno.

Questão 1		
	Freqüência	%
Boa	52	62,00
Regular	32	38,00
Ruim	0	0

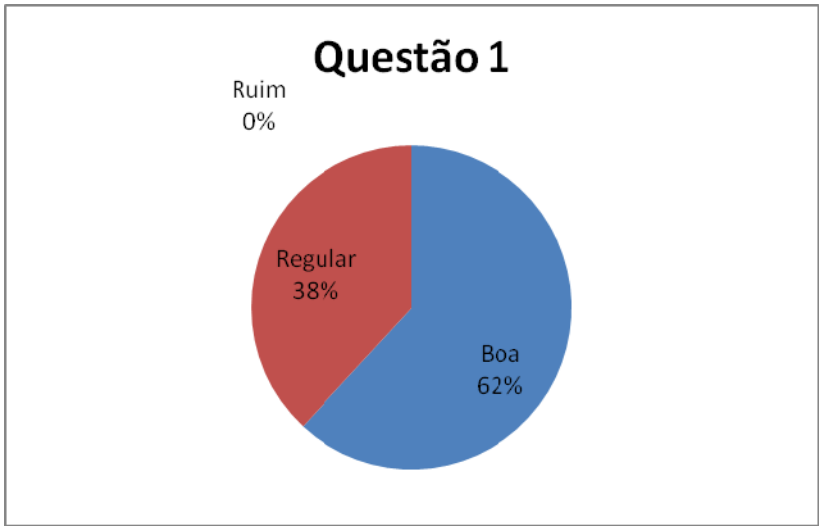

Figura 22 – Gráfico da Questão 1 do Ator Aluno.

Tabela 12 - Questão 2 do Ator Aluno.

Questão 2		
	Freqüência	%
Boa	33	39,00
Regular	44	53,00
Ruim	7	8

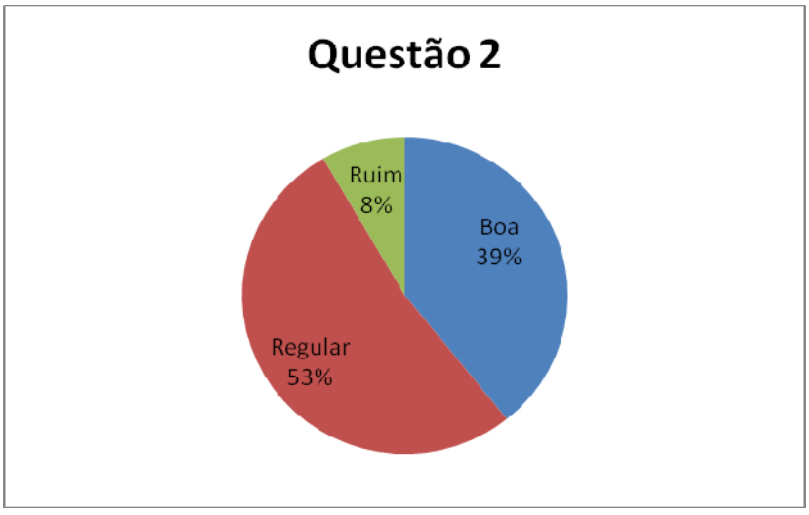

Figura 23 – Gráfico da Questão 2 do Ator Aluno.

Tabela 13 - Questão 3 do Ator Aluno.

Questão 3		
	Freqüência	%
Boa	64	76,00
Regular	20	24,00
Ruim	0	0

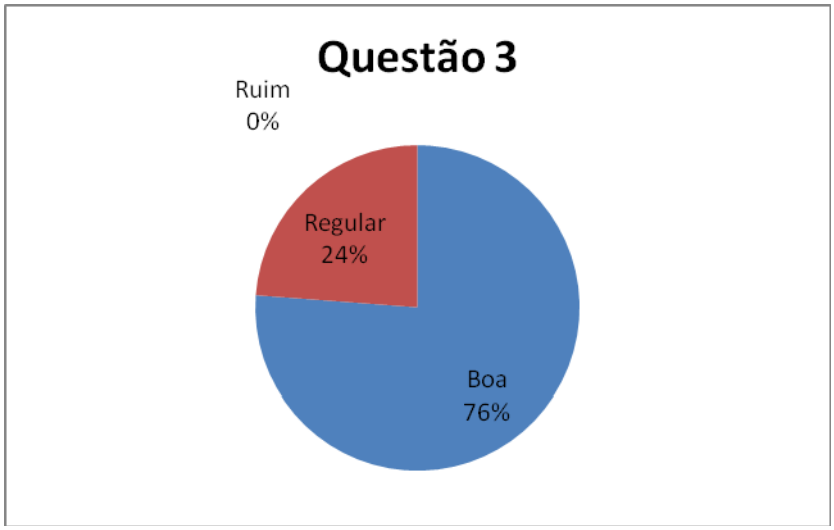

Figura 24 – Gráfico da Questão 3 do Ator Aluno.

Tabela 14 - Questão 4 do Ator Aluno.

Questão 4		
	Freqüência	%
Boa	52	61,00
Regular	31	38,00
Ruim	1	1

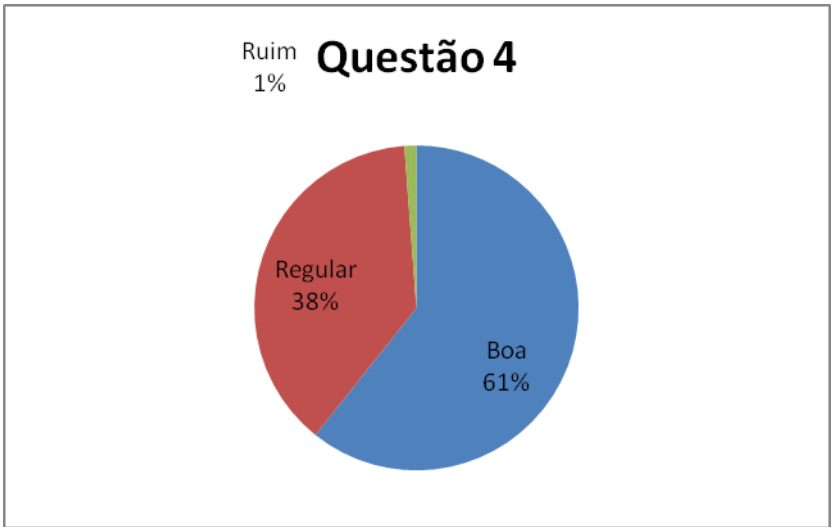

Figura 25 – Gráfico da Questão 4 do Ator Aluno.

Tabela 15 - Questão 5 do Ator Aluno.

Questão 5		
	Freqüência	%
Boa	54	64,00
Regular	30	36,00
Ruim	0	0

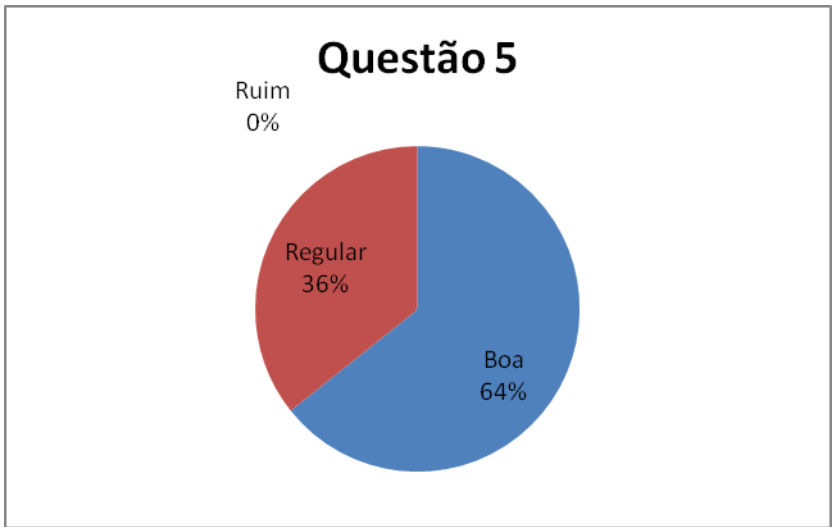

Figura 26 – Gráfico da Questão 5 do Ator Aluno.

Tabela 16 - Questão 6 do Ator Aluno.

Questão 6		
	Freqüência	%
Boa	42	50,00
Regular	42	50,00
Ruim	0	0

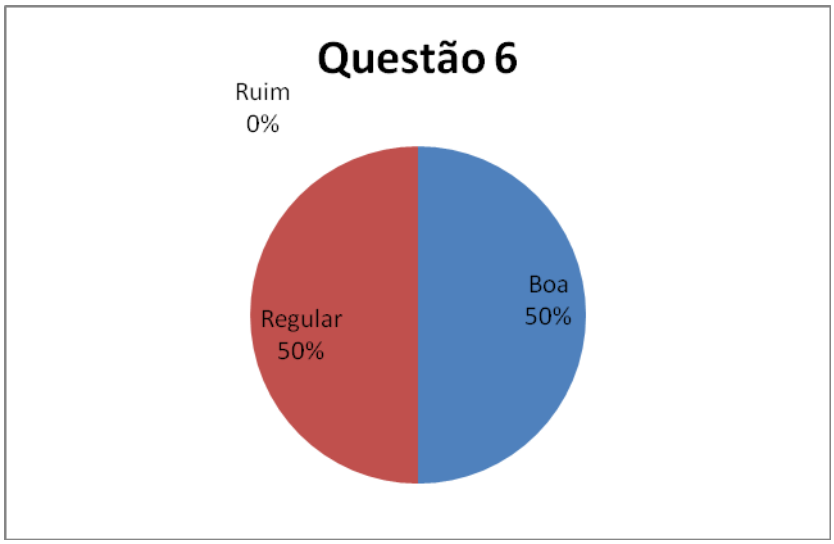

Figura 27 – Gráfico da Questão 6 do Ator Aluno.

Tabela 17 - Questão 7 do Ator Aluno.

Questão 7		
	Freqüência	%
Boa	68	81,00
Regular	16	19,00
Ruim	0	0

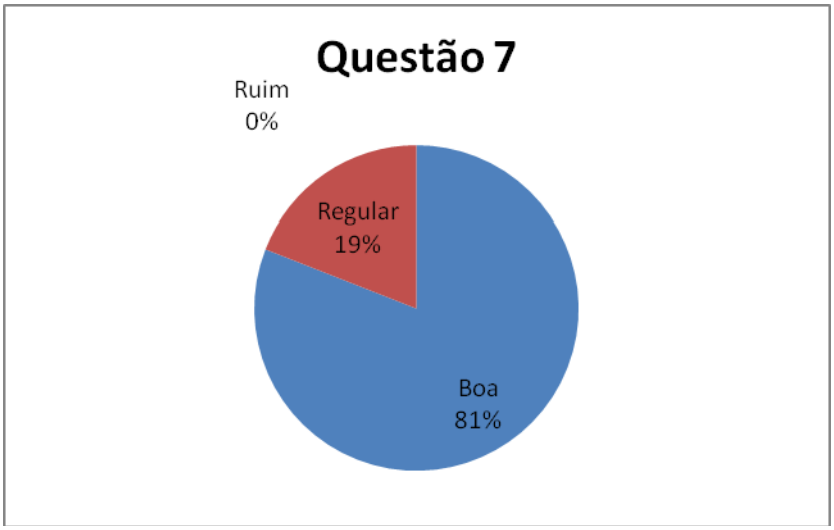

Figura 28 – Gráfico da Questão 7 do Ator Aluno.

Tabela 18 - Média do Ator Aluno.

MÉDIA	
Aluno 1	9,0
Aluno 2	10,0
Aluno 3	9,5
Aluno 4	9,0
Aluno 5	8,0
Aluno 6	10,0
Aluno 7	8,0
Aluno 8	9,0
Aluno 9	10,0
Aluno 10	9,0
Aluno 11	9,0
Aluno 12	10,0
Aluno 13	8,0
Aluno 14	9,0
Aluno 15	9,5
Aluno 16	9,0
Aluno 17	9,0
Aluno 18	9,0
Aluno 19	8,0
Aluno 20	8,0
Aluno 21	10,0
Aluno 22	8,0
Aluno 23	6,0

Aluno 24	9,0
Aluno 25	9,0
Aluno 26	8,0
Aluno 27	7,0
Aluno 28	8,0
Aluno 29	6,0
Aluno 30	10,0
Aluno 31	9,0
Aluno 32	8,5
Aluno 33	9,0
Aluno 34	8,0
Aluno 35	9,0
Aluno 36	8,0
Aluno 37	8,0
Aluno 38	5,0
Aluno 39	6,0
Aluno 40	8,0
Aluno 41	8,0
Aluno 42	9,0
Aluno 43	7,0
Aluno 44	7,0
Aluno 45	8,0
Aluno 46	8,0
Aluno 47	8,0
Aluno 48	9,0
Aluno 49	8,0
Aluno 50	8,0
Aluno 51	8,0
Aluno 52	9,0
Aluno 53	9,0
Aluno 54	10,0
Aluno 55	9,0
Aluno 56	8,0
Aluno 57	10,0
Aluno 58	9,0
Aluno 59	9,5
Aluno 60	9,0
Aluno 61	9,0
Aluno 62	8,5
Aluno 63	8,0
Aluno 64	10,0
Aluno 65	8,0
Aluno 66	9,0
Aluno 67	9,0
Aluno 68	6,0
Aluno 69	8,0
Aluno 70	8,0
Aluno 71	9,0
Aluno 72	8,0
Aluno 73	7,5
Aluno 74	7,0
Aluno 75	9,0

Aluno 76	8,0
Aluno 77	7,0
Aluno 78	9,0
Aluno 79	8,0
Aluno 80	7,0
Aluno 81	9,0
Aluno 82	8,0
Aluno 83	8,0
Aluno 84	9,0
Média	8,416667
Média Arredondada	8,42

Anexo III:

CONTEÚDO DO CD QUE ACOMPANHA ESTA DISSERTAÇÃO

O referido CD será composto pelo texto da dissertação em formato PDF, pelos arquivos executáveis, pelo formulário de testes para professores e alunos em formato PDF, pela *Java Virtual Machine* (JVM), pelos simuladores Scilab, Estabilidade Virtual, pelo leitor de arquivos PDFs Acrobat Reader, bem como pelo artigo da CALDIS, defendido na *International Conference on Signals and Electronic Systems*, realizada na *Poznań University of Technology Institute of Electronics and Telecommunications,* em 13-15 de setembro de 2004.