UNIVERSIDADE FEDERAL DO PARÁ
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

LABEXP – LABORATÓRIO DE EXPERIMENTAÇÃO REMOTA EM TEMPO REAL

Dissertação submetida ao PPGEE da Universidade Federal do Pará para a obtenção do Grau de Mestre em Engenharia Elétrica

DIEGO LIMA SANTOS

Belém, Agosto/2009

DIEGO LIMA SANTOS

LABEXP – LABORATÓRIO DE EXPERIMENTAÇÃO REMOTA EM TEMPO REAL

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Pará, como requisito para obtenção do Título de Mestre em Engenharia Elétrica.

Orientador: Dr. Carlos Tavares da Costa Jr.

Belém, Agosto/2009

DIEGO LIMA SANTOS

LABEXP – LABORATÓRIO DE EXPERIMENTAÇÃO REMOTA EM TEMPO REAL

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Pará, como requisito para obtenção do Título de Mestre em Engenharia Elétrica.

Data da defesa: 14/08/2009

_____
Prof. Dr. Marcus Vinícius Alves Nunes
Coordenador do Programa de Pós Graduação em Engenharia Elétrica

Banca Examinadora:

_____
Prof. Dr. Carlos Tavares da Costa Jr.
Orientador – Programa de Pós-Graduação em Engenharia Elétrica

_____
Prof. Dr. Orlando Fonseca Silva
Membro – Faculdade de Engenharia Elétrica

_____
Prof. Dra. Adriana Rosa Garcez Castro
Membro – Programa de Pós-Graduação em Engenharia Elétrica

_____
Prof. Dr. Dionne Cavalcante Monteiro
Membro – Programa de Pós-Graduação em Ciências da Computação

Dedico este trabalho a meus pais, meu irmão
e minha irmã, pelo apoio e incentivo.
Dedico também especialmente a Érica, por
estar sempre ao meu lado, incondicionalmente.

AGRADECIMENTOS

Ao professor Tavares pela oportunidade e confiança.

Ao professor Orlando, conhecido Nick, por sua orientação e incentivo.

Aos alunos do Programa de Educação Tutorial em Engenharia Elétrica (PET-EE), por ajudarem na aquisição de dados para esta dissertação, através da utilização do laboratório de experimentação remota.

Aos amigos do Laboratório de Recursos Hídricos, que também contribuíram para aquisição dos resultados.

A família e amigos pelo apoio e companheirismo em todos os momentos.

A minha namorada, Érica, pela compreensão, apoio, amizade e carinho durante o desenvolvimento deste trabalho e em toda minha vida.

A CAPES pelo apoio financeiro.

Embora ninguém possa voltar atrás
e fazer um novo começo,
Qualquer Um pode Começar agora
e fazer um Novo Fim.(Chico Xavier).

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Este trabalho apresenta o processo de desenvolvimento e implementação do Laboratório de Experimentação Remota em Tempo Real (LabExp), atualmente em funcionamento na Universidade Federal do Pará, com o objetivo de funcionar como uma plataforma auxiliar para ensino e aprendizagem das disciplinas de sistemas de controle. O ensino e aprendizagem foram contemplados através da disponibilização de experimentos, onde os usuários poderão interagir com os mesmos, alterando parâmetros e observando o resultado desta interação.

Além dos experimentos disponíveis, acredita-se que em ambientes de educação online é interessante disponibilizar aos alunos ferramentas que proporcionem maior interação entre alunos e professores e com o próprio laboratório remoto, proporcionando uma metodologia de aprendizado mais colaborativo, estimulando o aluno. Desta forma, são disponibilizadas aos alunos três aplicações: uma para envio de seus próprios experimentos; outra para interação com outros alunos, através de um fórum; e outra para o envio de suas opiniões/críticas.

Antes do processo de desenvolvimento e implementação do LabExp, foi realizada uma análise sucinta sobre educação online, tendo em vista ser esta a finalidade do laboratório. Esta análise proporcionou maior conhecimento sobre esta metodologia de educação, orientando no restante do desenvolvimento do LabExp.

Compreende-se que as tecnologias utilizadas não são determinantes para o desenvolvimento de um laboratório remoto, voltado para a educação online, entretanto experimentos remotos de sistemas de controle possuem uma restrição temporal, ou seja, necessitam obedecer a limites de tempo restritos, funcionando em tempo real. Para conseguir este comportamento foi utilizada a Real-Time Application Interface (RTAI), com o componente RTAI-XML.

Além das tecnologias utilizadas, neste trabalho também é apresentado o processo de modelagem do LabExp, de acordo com padrões, princípios e recursos da Unified Modeling Language (UML) aplicada a aplicações web. Este processo de modelagem foi de fundamental importância, pois facilitou e orientou o desenvolvimento do laboratório.

Palavras-Chave: Experimentação Remota, Sistemas de Controle, Educação Online, Modelagem UML, Sistemas em Tempo Real.

ABSTRACT

This work proposes the Real Time Remote Experimentation Laboratory's (LabExp) development and implementation process. This laboratory is actually working in Universidade Federal do Pará aiming to act as an auxiliary tool for teaching and learning of control systems disciplines. Teaching and learning were included through experiments availability, where users may interact with them, changing parameters and observing this interaction results.

Besides experiments availability, it is believed that in online learning environments it is interesting to make available for students tools that provide greater interaction between students and teachers and with the remote laboratory, providing a more collaborative approach to learning, encouraging the student. This way, it is available to students three applications: one to submit their own experiments, another for interaction with other students through a forum, and another to send your opinions/criticisms.

Before LabExp's development and implementation process, it was make a brief analysis about online education, because this is the laboratory's purpose. This analysis provided more knowledge about this method of education, guiding the rest of LabExp's development.

It is understood that the technologies used are not critical for a remote laboratory's development, directed to online education, however remote experiments of control systems have a temporal restriction, i.e. need to obey restrict time limits, working in real time. To achieve this behavior it was used the Real Time Application Interface (RTAI), with the RTAI-XML component.

Beyond used technologies, this work also presents LabExp's modeling process, according to Unified Modeling Language (UML) standard, principles and resources, applied to web applications. This modeling process was of fundamental importance, because it facilitated and guided the laboratory's development.

Key-words: Remote Experimentation, Control Systems, Online learning, UML Modeling, Real Time Systems.

1 INTRODUÇÃO

As discussões a respeito da utilização de tecnologia para estimular e complementar a educação ainda é um assunto bastante referenciado. Pesquisas mostraram que estudantes adquirem benefícios significativos no aprendizado quando utilizam mídias áudio-visuais ou computadores, além do método tradicional (CLARK, 1983). Entretanto esta mesma pesquisa sugeriu que a razão para esses benefícios não é o meio de instrução, mas a estratégia instrucional construída nos materiais de aprendizado. Assim, pode-se dizer que o aprendizado é mais influenciado pelo conteúdo e pela estratégia instrucional, utilizada nos materiais de aprendizado, que pelo tipo de tecnologia utilizada para transmitir a instrução.

Sendo assim, deve-se haver uma preocupação adicional no modo como as novas tecnologias são utilizadas para transmitir conhecimento. Atualmente observou-se que a internet é uma tecnologia que vem sendo bastante utilizada no meio educacional e através da sua popularização foi possível se perceber uma nova estratégia instrucional para transmissão do conhecimento, como em (HETZOG et al., 2005) e (FORTE et al., 2008). Através da internet usuários podem acessar uma quantidade ilimitada de informações; acessar conteúdo a qualquer hora e de qualquer lugar; possuem facilidade de comunicação, inclusive instantânea; e podem compartilhar recursos.

Além da popularização da internet foi possível acompanhar em épocas recentes a consolidação de microcomputadores com distintas finalidades, envolvendo desde simples tarefas até complexos controles de processos industriais. Esta consolidação deve-se, em parte, a utilização de redes de computadores e de novas tecnologias que surgem diariamente na tentativa de automatizar processos simples ou complexos.

Uma das possibilidades da utilização de microcomputadores no meio educacional é através de simulações computacionais, principalmente nas áreas de sistemas de controle. As demonstrações baseadas em simulações em computador, localizados localmente ou remotamente, de uma maneira geral, tornam os alunos mais participativos. A possibilidade de rapidamente mudar parâmetros e verificar a conseqüência nos movimentos estudados, que uma simulação computacional proporciona, pode estimular os estudantes a querer se aprofundar sobre o funcionamento do sistema nas mais diversas situações. Este tipo de metodologia, mais interativa, é um dos elementos que poderá tornar o processo de ensino mais dinâmico (SCHUMACHER et al., 2004).

A utilização da internet e de microcomputadores, voltados para a educação, são base de uma nova metodologia de transmissão do conhecimento: a educação online. Diferentes terminologias têm sido utilizadas para educação online. Termos que são comumente utilizados são *e-leaning, tele-learning*, aprendizado virtual e educação à distância. Todos estes termos implicam que um estudante está distante de seu professor ou instrutor, que utiliza algum tipo de tecnologia (geralmente o computador) para acessar materiais de aprendizado, que utiliza a tecnologia para interagir com o instrutor ou com outros estudantes, e que os estudantes possuem algum tipo de suporte. Este trabalho adotará a terminologia educação online.

Cada vez mais empresas e instituições de ensino estão adotando a educação online como a principal metodologia para treinar funcionários e educar alunos, respectivamente (SIMMONS, 2002). Para estas empresas e instituições existem alguns benefícios tanto para instrutores quanto para aprendizes. Para aprendizes não existem barreiras de tempo, espaço e distância na educação online. Na educação online assíncrona estudantes podem acessar os materiais online a qualquer hora, enquanto que a síncrona permite interações em tempo real entre estudantes e instrutores. Estudantes podem utilizar a internet para acessar materiais de aprendizado atualizados e relevantes e podem se comunicar com especialistas em determinada área. Além disso, os estudantes podem completar cursos online enquanto estão trabalhando em suas empresas ou em sua própria casa.

Para o instrutor o ensino pode ser realizado em qualquer hora e em qualquer lugar. Materiais online podem ser atualizados e os estudantes podem acessá-los imediatamente. Se propriamente desenvolvidos, os sistemas de educação online podem ser utilizados para determinar as necessidades e o atual estado de conhecimento dos estudantes, e para determinar apropriadamente os materiais que os estudantes devem selecionar para atingir o resultado desejado.

Uma utilização específica da educação online em instituições de ensino refere-se às práticas de laboratório, comuns em diversas áreas. Laboratórios remotos virtuais utilizados em aulas práticas são uma forma didática recente, utilizada em diversas universidades distribuídas pelo mundo, como a Universidade Nacional de Singapura (HOON, 1998), Universidade de Padova (BERTOCCO et al., 1998), Universidade do Estado de Portland (STEGAWSKI; SCHAUMANN, 1998), Universidade Politécnica de Valência (PALOP; TERUEL, 2000), Universidade de São Paulo (MOSSIN, 2007), Escola Politécnica Federal de Lausanne (NGUYEN et al., 2005), Universidade de Florença (BASSO; BAGNI, 2004), Universidade de

Siena (CASINI et al., 2004) e a Universidade de Tecnologia de Helsink (POHJOLA, 2006). Nestas instituições são disponibilizados experimentos, reais ou simulados, para interação remota, ou seja, através da internet. Este paradigma de ensino foi denominado experimentação remota.

Além da utilização em instituições de ensino, laboratórios remotos têm sido utilizados em outras áreas do conhecimento. Em (MAGRABI et al., 1999) é apresentado um sistema de tele-medicina onde se podem acompanhar os batimentos cardíacos de um paciente, através da internet. Em (PLOTHEGER; FERNANDES, 2005), é apresentado um Sistema de Diagnóstico Remoto (SDR) para ambiente industrial aplicado em máquinas-ferramenta (MF). Em (NETO; TERUEL, 2008) é apresentado o TERMICONT, um software para supervisão e controle de uma planta de secagem, monitoramento da temperatura e umidade do ar, velocidade de rotação do motor de um ventilador e consumo de energia, sendo este aplicativo acessado tanto local, quanto remotamente (através de um laboratório remoto), controlando a vazão de ar e potência da resistência de aquecimento e realizando a aquisição e armazenamento dos dados do processo.

Particularmente se tratando de sistemas de controle compreende-se que é fundamental para os alunos aplicar na prática os conhecimentos teóricos adquiridos. Entretanto, os materiais necessários para realização de experiências deste tipo são de custo elevado. Sendo assim, acredita-se que para um país com as dimensões territoriais e as desigualdades econômicas e sociais do Brasil, deve-se pensar em alternativas para a metodologia tradicional de ensino, combinando ensino presencial e online. Compreende-se, também, que manter uma instituição de ensino superior, seja pública ou privada, requer alto custo financeiro, por isso, para alguns cursos, a combinação entre estas duas modalidades de ensino pode ser considerada uma opção interessante. Assim, os alunos poderiam aprender a teoria em sala de aula e, remotamente, acessar as experiências e por em prática o conhecimento adquirido, independente da localização desta experiência.

O objetivo deste trabalho é apresentar o processo de desenvolvimento e implementação do Laboratório de Experimentação em Tempo Real (LabExp), atualmente em funcionamento na Universidade Federal do Pará. Este laboratório foi desenvolvido almejando funcionar como uma ferramenta auxiliar para o ensino-aprendizagem de sistemas de controle. Para realizar esta implementação, os seguintes objetivos secundários foram considerados:

- Utilização de um sistema operacional de tempo real. Compreende-se que para o desenvolvimento de um laboratório de experimentação remota a tecnologia não deve

ser o fator determinante, entretanto experimentos de sistemas de controle possuem uma restrição tecnológica, pois necessitam atuar em limites de tempo específicos, ou seja, em tempo real.

- Proporcionar uma forma dinâmica de aprendizado online. Através da utilização de aplicações, como fórum, envio de arquivos e questionários, pretende-se proporcionar uma metodologia de aprendizado online mais colaborativo, estimulando a participação dos alunos;
- Modelagem do Laboratório. Por se tratar de uma aplicação web, as interfaces estáticas e dinâmicas do LabExp foram desenvolvidas de acordo com padrões, princípios e recursos da *Unified Modeling Language* (BOOCH et al., 1997) aplicada a aplicações web, facilitando e orientando seu desenvolvimento.

Este trabalho está estruturado do seguinte modo, além deste capítulo introdutório: no capítulo 2 são abordados conceitos sobre a educação online considerados importantes para este trabalho. No capítulo 3 é realizada a análise das possíveis tecnologias disponíveis para o desenvolvimento de um laboratório de experimentação remota, independente da finalidade. As tecnologias utilizadas no LabExp são apresentadas no capítulo 4. No capítulo 5 é apresentada a modelagem e desenvolvimento das interfaces do LabExp, sendo este considerado uma aplicação web. No capítulo 6 são apresentados os resultado obtidos. Finalmente no capítulo 7 são apresentadas as conclusões e propostas futuras.

2 EDUCAÇÃO ONLINE

O processo de desenvolvimento do Laboratório de Experimentação Remota em Tempo Real (LabExp) consistiu na definição da sua concepção; definição das tecnologias a serem utilizadas; modelagem do laboratório, sendo este considerado como uma aplicação web; implementação das tecnologias; e desenvolvimento das interfaces.

A concepção do LabExp foi atender estudantes das áreas de sistemas de controle, funcionando como uma ferramenta auxiliar adicional para professores e alunos. Neste laboratório é permitido que os estudantes interajam com experimentos e enviem seus próprios experimentos, sendo desenvolvidos de acordo com algumas especificações; troquem informações com outros estudantes e/ou tutores; e emitam opiniões pertinentes sobre a utilização do laboratório, desenvolvendo conhecimento através de aprendizado colaborativo.

Após a definição da concepção do LabExp, foi necessário analisar quais tecnologias estariam disponíveis atualmente, com o objetivo de definir quais seriam mais viáveis para atender esta concepção. Entretanto antes de analisar estas tecnologias, acreditou-se que seria interessante se realizarem algumas considerações sobre educação online, pois esta modalidade de educação é utilizada por este laboratório.

Compreende-se que atualmente existem diversos laboratórios de experimentação remota, distribuídos mundialmente, com propósitos e finalidades distintos. A utilização de um laboratório de experimentação remota voltado para educação deve ser analisada através de uma abordagem específica, obedecendo a requisitos relacionados à educação online. Educação online, *e-learning*, aprendizado através da internet, educação à distância (EAD), educação virtual, entre outras terminologias, caracterizam a metodologia de aprendizado em que o aluno permanece distante do professor ou instrutor, utilizando o computador para acessar materiais e receber suporte/orientação. Nesta dissertação será utilizada a terminologia educação online para este fim. Existem, atualmente, distintos conceitos para educação online, contudo, concorda-se com (ALLY, 2004) ao definir como:

"A utilização de alguma forma de tecnologia para acessar materiais de aprendizado; para interagir com conteúdo, instrutor e outros alunos; e obter suporte durante o processo de aprendizado, almejando adquirir conhecimento, para construir conhecimento pessoal e para desenvolvimento através da experiência de aprendizado."

Na educação online, a disponibilidade de tecnologia não deve ser o fator determinante, mas a disponibilidade de pessoal qualificado para atuar ativamente neste ambiente, o desenvolvimento de mecanismos que proporcionem a participação de alunos de modo colaborativo (ferramentas multimídia, objetos de aprendizado), a análise de aspectos referentes ao desenvolvimento do web site (design visual, design instrucional), entre outros aspectos, também devem ser analisados. A situação real, entretanto, é menos racional. Iniciativas de educação online freqüentemente nascem da experimentação individual de um educador ou um pequeno grupo de educadores ou tecnólogos que não tem idéia do beneficio que o experimento irá trazer para a experiência de educação, mas que são bem intencionados. Mesmo onde o mercado de estudantes é bem compreendido e os resultados do aprendizado são claramente definidos, a implementação da educação online freqüentemente envolve uma boa quantidade de tentativa e erro. Por esta razão acreditou-se ser fundamental realizar uma análise sobre os aspectos referentes à educação online.

Neste capítulo serão abordados alguns aspectos da educação online, com o objetivo de evidenciar os que foram considerados no desenvolvimento do LabExp, do ponto de vista educacional.

Na seção 2.1 será apresentado o histórico da EAD a nível mundial e no Brasil. Na seção 2.2 será discutida a utilização da internet como meio de educação. Na seção 2.3 alguns fundamentos básicos da educação online serão abordados. Em seguida algumas considerações sobre ensino presencial e ensino a distância serão discutidas. Finalizando este capítulo dois laboratórios de experimentação remota, voltados para educação online, são apresentados.

2.1 HISTÓRICO EAD

Segundo (ALVES, 1998), a EAD teve inicio no século XV, quando Johannes Guttenberg, na Alemanha, inventou a imprensa, com composição de palavras por caracteres móveis. Naquele momento a produção de livros possuía custo muito elevado, pois estes eram produzidos manualmente, dificultando o acesso ao conhecimento. (ALVES, 1998) também cita que a difusão da EAD no mundo deve-se principalmente a França, Espanha e Inglaterra, pois os centros educacionais destes países contribuíram para que outros pudessem adotar os modelos desenvolvidos.

Em épocas mais recentes, em 1880, na Inglaterra foi criado um curso de correspondência, com direito a diploma, sendo, entretanto, não reconhecido pelas autoridades locais, fazendo com que seus autores se mudassem para os Estados Unidos, onde encontraram reconhecimento na Universidade de Chicago. Em 1882, surgiu, naquela instituição, o primeiro curso por correspondência, com material enviado pelo correio (NISKIER, 1999).

No final do século XX foi criada a Open University (OU), na Inglaterra, sendo a primeira universidade baseada totalmente no conceito de EAD. Seus cursos iniciaram em 1970, e, em 1980, já possuíam 70 mil alunos. Ao longo de sua existência, foram incorporadas todas as novas tecnologias que eram desenvolvidas e popularizadas, como vídeos e computadores pessoais na década de 80, e a internet na década de 90 (NISKIER, 1999).

Vários países desenvolveram sistemas de EAD para lidar com condições específicas que apresentavam desafios para o processo de educação tradicional. Países como o Canadá, Suécia, Dinamarca, Noruega e Finlândia possuem regiões geladas durante maior parte do ano, com acesso terrestre muito difícil. Outra condição que levou alguns países a desenvolverem mecanismos de EAD são as vastas extensões geográficas. Exemplos de países que se enquadraram nesta situação são Austrália e, novamente, Canadá. Os países, citados como exemplo acima, desenvolveram a EAD a partir dos anos de 1910/20 (NISKIER, 1999).

No Brasil, apesar da inexistência de fatos precisos, o Jornal do Brasil registrou na primeira seção de classificados anuncio oferecendo profissionalização por correspondência (datilógrafo), em 1891. Em 1923, com a fundação da Rádio Sociedade do Rio de Janeiro, iniciou-se o processo de educação pelo rádio. Em 1939, em São Paulo, o Instituto de Rádio Técnico Monitor, ofereceu o curso de eletrônica a distância. Em 1946, o Serviço Nacional de Aprendizagem Comercial (SENAC), desenvolveu, no Rio de Janeiro e São Paulo, a Universidade do Ar, atingindo, em 1950, 380 localidades e 80 alunos, e, em 1973, iniciou cursos de correspondência, seguindo o modelo da Universidade de Wisconsin, Estados Unidos. No fim da década de 80 e inicio dos anos 90, notou-se grande avanço da EAD brasileira, havendo grande número de cursos através de vídeos e fitas K-7, como formas de auto-aprendizagem (ALVES, 1998).

Atualmente, no Brasil, existem cursos de EAD para ensino médio, graduação e pós-graduação, em diversas áreas do conhecimento, reconhecidos pelo Ministério da Educação (MEC). O crescimento da EAD pode ser visto nos resultados do Censo de Educação Superior de 2006, divulgado pelo Instituto Nacional de Estudos e Pesquisas Educacionais (INEP), observando-se que a oferta de cursos superiores de EAD cresceu 571% entre 2003 e 2006,

passando de 52 para 349. A participação destes alunos no universo dos estudantes passou a ser de 4,4% em 2006, sendo que, um ano antes, essa participação representava 2,6%.

Diversas tecnologias modernas podem ser utilizadas para proporcionar a educação online, como internet e vídeo aulas, por exemplo. Entretanto um bom curso de ensino a distância deve ser desenvolvido de acordo com alguns requisitos, pois, conforme especificado anteriormente a tecnologia disponível não deve ser o fator fundamental. Na próxima seção serão apresentadas características sobre a internet, e, em seguida, considerações sobre estes requisitos são apresentadas.

2.2 CARACTERÍSTICAS DA INTERNET PARA EDUCAÇÃO ONLINE

A World Wide Web, ou simplesmente internet, é uma tecnologia distinta que provê várias ferramentas para comunicação e informação que podem ser utilizadas para educação. Entretanto existem obstáculos que precisam ser superados para a utilização destas ferramentas. O acesso majoritário a internet acontece em casa ou no trabalho, contudo, em países desenvolvidos, bibliotecas públicas e cyber cafés, além de conexões através de dispositivos de conexão *wireless* (sem-fio) fazem com que o acesso não apresente problemas físicos para a maioria da população. Em países subdesenvolvidos, como o Brasil, o acesso ainda é problemático para aqueles que possuem limitações físicas, ou seja, o usuário que não disponibilizar de acesso a internet dificilmente poderá contar com a tecnologia *wireless*, pois esta ainda não é utilizada em larga escala e não possui grande disponibilidade quanto nos países desenvolvidos.

O acesso a internet está aumentando, não somente para tecnologia, mas também em outras áreas, como: jornais escolares (E-JOURNALS, 2009), objetos de aprendizado (MERLOT, 2009) e referências para milhões de páginas com conteúdo comercial, educacional e cultural (GOOGLE, 2009). Assim, a teoria de educação online deve reconhecer a mudança de uma era de conteúdos restritos e limitados para uma era em que a quantidade de conteúdo é tão grande que a filtragem e redução de conteúdo são tão importantes quanto prover conteúdo suficiente. Além disto, esta nova era também é caracterizada pela mudança da apresentação de conteúdos somente em textos para interações com todas as formas de

mídia, como vídeo *streaming*, tele e áudio conferência, e utilização de mundos virtuais, como o Second Life (SECOND LIFE, 2009), por exemplo.

Outro fator relevante para o crescimento do acesso a internet é a facilidade com que conteúdos podem ser atualizados e revisados. Por exemplo, a utilização de Web blogs e sistemas para gerenciamento do aprendizado, como TelEduc (TELEDUC, 2009) e Moodle (MOODLE, 2009), criam ambientes onde professores e alunos podem criar e atualizar seus conteúdos sem a ajuda de programadores ou designers. Entretanto, a falta de ajuda destes especialistas pode ocasionar em erros e geração de conteúdos sem um padrão profissional (ANDERSON, 2004).

A utilização da internet como meio para a educação online pode ser abordada de modos distintos: transmissão de aulas ao vivo aos alunos, através da implementação ou não de uma Rede Virtual Privada (*Virtual Private Network* - VPN); disponibilização de vídeo aulas para download em algum ambiente; e elaboração de uma ambiente colaborativo, onde os alunos participariam da criação do conteúdo, entre outras possibilidades. Nesta dissertação foi utilizada esta última abordagem para desenvolvimento do LabExp.

Ressalta-se, entretanto, que o computador não faz com que os alunos aprendam, mas, sim, o desenvolvimento de modelos da vida real e simulações e a interação dos alunos com estes modelos e simulações (CLARK, 1983).

2.3 FUNDAMENTOS DA EDUCAÇÃO ONLINE

Atualmente muitas empresas e instituições de ensino estão utilizando a internet como meio para ensino/treinamento. Os benefícios deste processo de migração podem ser observados tanto para alunos/empregados quanto para instrutores. Através da educação online, os alunos não encontram barreiras de tempo e não existem problemas relacionados à localização e distância. Além disto, no aprendizado online assíncrono, alunos podem acessar o material em qualquer horário, enquanto o aprendizado síncrono permite interação em tempo real entre instrutor e aluno; alunos podem utilizar a internet para obter materiais atualizados e relevantes, além de permitir a comunicação com especialistas no campo de estudo; podem-se completar cursos online enquanto estiverem trabalhando em seu emprego. Para o instrutor, o

processo de instrução pode ser realizado a qualquer hora e de qualquer lugar. O instrutor pode atualizar materiais online, permitindo aos alunos obtê-los imediatamente.

Sendo assim, apesar do desenvolvimento de novos recursos tecnológicos criados visando facilitar o processo de aprendizagem, deve haver uma preocupação especial quanto à apresentação de material multimídia, pois sua má implementação pode resultar em desorientação e desmotivação do usuário. Utilizando-se imagens, sons e experiências de simulação e experimentação, a atividade multimídia pode envolver o usuário de modo que o mantenha sempre interessado em utilizar o ambiente.

Por isso, acredita-se que, para utilização de laboratórios virtuais remotos como ambiente colaborativo de aprendizado, alguns princípios básicos de design devem ser analisados (NASCIMENTO, 2005):

- Características do usuário: antes de iniciar o desenvolvimento do ambiente, faz-se necessário conhecer as características dos possíveis usuários deste ambiente, visando elaborar as melhores estratégias pedagógicas quanto aos elementos multimídia;
- Planejamento da interface instrucional: a interface deverá ser consistente e agradável do ponto de vista estético, a fim de orientar e atrair o usuário. A informação apresentada em uma página html deve ser organizada de forma a facilitar a sua visualização e interpretação correta;
- Navegação: é importante ajudar o usuário a se orientar usando técnicas para orientar sua atual localização;
- Elementos multimídia: a utilização de múltiplos formatos de informação (simulações, imagens estáticas, textos, som, animações, vídeos) desempenha um papel importante na aquisição do conhecimento quando bem utilizados. Particularmente tratando de simulações, pode-se oferecer ao usuário um laboratório que possibilita a aprendizagem por análise das reações de causa e efeito, oferecendo feedback em tempo real aos usuários;

2.4 ENSINO PRESENCIAL X DISTÂNCIA

A presença de um professor/tutor durante o processo inicial de aprendizagem faz-se fundamental, não somente para lecionar e avaliar o desenvolvimento do aluno, mas para

perceber a motivação do aluno visando mudar ou adaptar sua metodologia de ensino. Este processo inicial de aprendizagem pode ser caracterizado na graduação ou ensino médio, dependendo do nível de maturidade e autonomia deste aluno. Na tabela 1, são apresentas algumas vantagens e desvantagens da educação online (COSTAS, 2006).

Tabela 1

Vantagens e Desvantagens da Educação Online

Vantagens	Desvantagens
Flexibilidade: o conteúdo das aulas pode ser acessado em qualquer horário.	Solidão: o contato com outros alunos limita-se aos debates na internet e é mais difícil fazer amizades.
Fim da barreira geográfica: pode-se estudar na melhores universidades, mesmo morando em regiões distantes.	Excesso de independência: é preciso ter disciplina; não há o estimulo nem a supervisão do professor.
É mais barato: se gasta menos com mensalidade e transporte.	Falta uma biblioteca: o catalogo de livros digitalizados na internet é pequeno comparado com o das bibliotecas de faculdades.

Fonte: (COSTAS, 2006)

Acredita-se que para um país com as dimensões territoriais e as desigualdades econômicas e sociais do Brasil, deve-se pensar na combinação entre ensino presencial e a distância, pois se compreende que manter uma instituição de ensino superior, seja pública ou privada, requer alto custo financeiro e, por isso, para alguns cursos, a combinação entre estas duas modalidades de ensino pode ser considerada uma opção interessante. Utilizando como exemplo um curso de sistemas de controle, faz-se necessário que o aluno tenha contato direto com experimentos visando aprender, na prática, os conceitos teóricos aprendidos em sala de aula. Entretanto, os materiais necessários para realização de experiências deste tipo são de custo elevado. Assim, os alunos poderiam aprender a teoria em sala de aula e, através da internet, acessar as experiências e por em prática o conhecimento adquirido, independente da localização desta experiência.

Relacionado aos custos financeiros envolvidos no desenvolvimento de um laboratório remoto, ressalta-se que um planejamento bem elaborado deve ser realizado, senão estes custos

poderão ser bastante elevados. No processo de elaboração do ambiente web onde os alunos poderão interagir com as experiências e trocar informações entre si e com professores, faz-se necessário o trabalho de vários profissionais: web designer, especialista em ensino online, digitadores, além do próprio professor. Logo, cada um destes profissionais demandará um custo adicional para instituição. Entretanto, cada profissional pode desenvolver vários ambientes, por isso, quanto mais turmas houver, menor será o custo (BASTOS, 2001).

Outro aspecto importante que vale destacar na educação online é a interatividade. A internet permite a interação em diversas modalidades. Na figura 1, pode-se observar as formas comuns de mídia utilizadas na educação, representando graficamente suas capacidades de independência de tempo e lugar versus sua capacidade de prover interação.

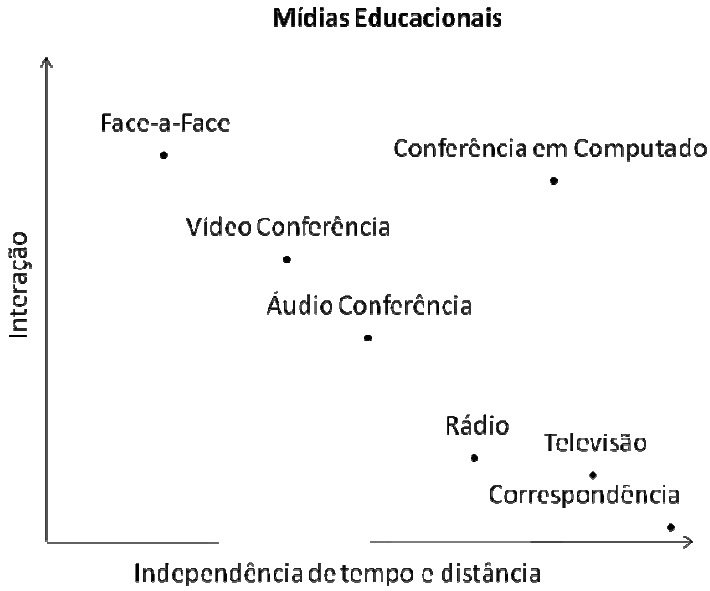

Figura 1 Atributos das mídias educacionais
Fonte: (ANDERSON, 2004)

Pode-se observar que quanto maior e mais rica a forma de interação, maiores serão as restrições de tempo e lugar. Na figura 2 pode-se observar a capacidade que a internet possui de dar suporte a estas modalidades.

Figura 2 Mídias educacionais suportadas pela Web
Fonte: (ANDERSON, 2004)

Conforme pode ser observado, todas as formas de mídias educacionais de interação são agora suportadas, acreditando-se na adição da utilização da internet para melhorar a educação baseada em salas de aula. Assim, a capacidade que a Web possui de dar suporte a educação online é geralmente um grande domínio para discussões significantes até que se especifique a modalidade particular de interação a ser utilizada.

A interatividade pode ser delineada no contexto dos atores envolvidos. Os atores envolvidos no processo de educação online são os alunos, os professores e o conteúdo utilizado. Sendo assim, possuímos 06 tipos de interação: aluno-aluno, aluno-conteúdo, aluno-professor, professor-professor, professor-conteúdo e conteúdo-conteúdo (ANDERSON, 2004).

A interação aluno-aluno é um requisito fundamental para educação online. Através desta interação, alunos podem investigar e desenvolver múltiplas perspectivas de um mesmo problema; podem trabalhar através de aprendizado colaborativo, aumentando a aquisição de habilidades sociais na educação; e podem desenvolver habilidades de comunidades de aprendizado (WENGER et al., 2002), permitindo aos alunos desenvolver habilidades interpessoais e investigar o conhecimento compartilhado por membros da comunidade.

A interação entre professor e aluno é obtida, na educação online, de diversos modos e formatos, que incluem comunicação síncrona e assíncrona utilizando texto, áudio e vídeo.

A Web atende a interação entre aluno e conteúdo provendo novas oportunidades quando comparada a metodologia tradicional de ensino, incluindo inserção em micro

ambientes, exercícios em laboratórios virtuais, tutoriais auxiliados por computador e o desenvolvimento de conteúdo interativo que responde ao comportamento e atributos do aluno.

A interação entre professores, no contexto da educação online, cria novas oportunidades para desenvolvimento e apoio profissional, através de comunidades, encorajando-os a obter vantagem do conhecimento desenvolvido e descoberto em sua área de conhecimento.

A interação entre professor e conteúdo foca na criação de conteúdo e atividades de aprendizado por professores, permitindo que estes possam continuamente atualizar e monitorar o material e as atividades desenvolvidas.

A interação conteúdo-conteúdo é um modo de educação online onde o conteúdo é programado para interagir com outras fontes automáticas de informação, tanto para se atualizar constantemente quanto para adquirir novas capacidades. Por exemplo, em um laboratório virtual de sistemas de controle, a interface de comunicação com o aluno interage com um servidor de experiências constantemente, obtendo e apresentando sempre a resposta atualizada deste servidor. A figura 3 apresenta um modelo de educação online, contendo as interações discutidas acima.

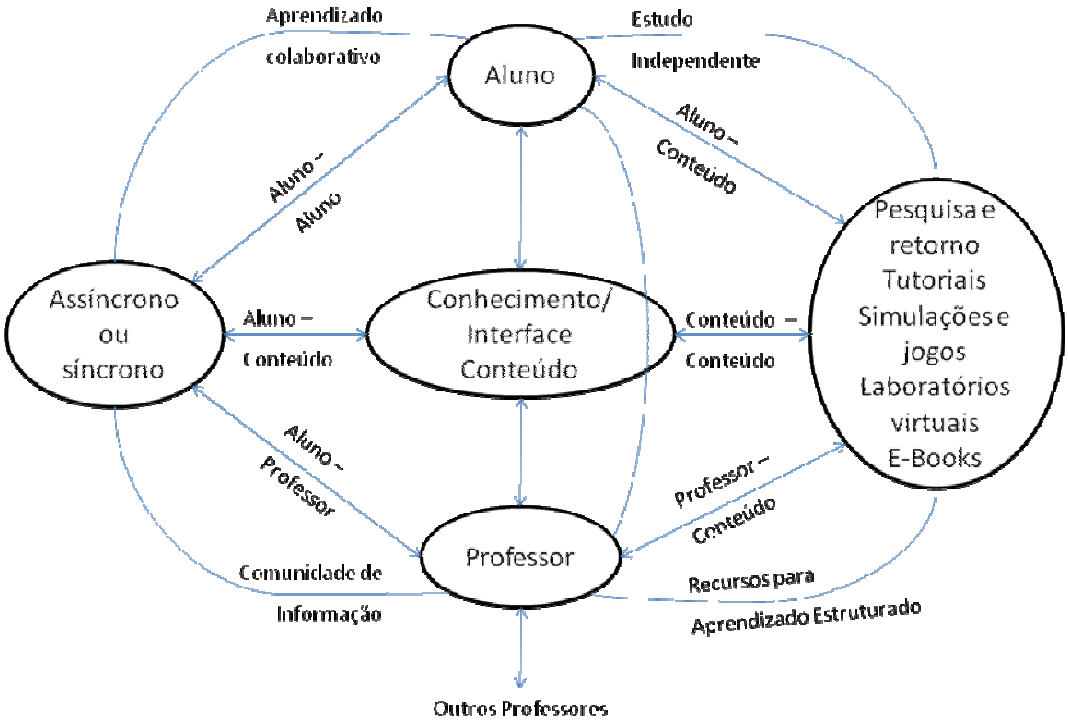

Figura 3 Modelo de educação online
Fonte: (ANDERSON, 2004)

Nesta figura pode-se observar a interação entre alunos e professores e a interação com conteúdo. Os alunos podem interagir diretamente com o conteúdo que é encontrado na internet, entretanto muitos preferem optar pelo aprendizado seqüenciado, direto e verificado por um professor ou tutor. Esta interação pode ocorrer através de uma comunidade de informação, utilizando uma variedade de atividades síncronas ou assíncronas baseadas na internet (vídeo, áudio, conferência, chats, fórum, ou interação de mundos virtuais). Estes ambientes são ricos e permitem o aprendizado de habilidades sociais, o aprendizado colaborativo de conteúdo e o desenvolvimento de relações interpessoais entre participantes.

Na figura 3 também são ilustradas as ferramentas de aprendizado associadas com aprendizado independente. Ferramentas comumente utilizadas são tutoriais, práticas e simulações. Laboratórios virtuais, onde alunos realizam simulações de experimentos de laboratórios, e sofisticadas ferramentas de pesquisa também estão se tornando instrumentos comuns para aprendizado individual. Vale ressaltar que, apesar da caracterização de estudo independente, o aluno não está sozinho. Freqüentemente colegas de trabalho, localizados localmente ou remotamente, tem se mostrado fontes significantes de suporte e assistência para alunos independentes.

2.5 LABORATÓRIOS REMOTOS UTILIZADOS PARA EDUCAÇÃO ONLINE

Conforme ratificado anteriormente, existem atualmente muitos laboratórios remotos, com objetivos distintos. Acredita-se, porém, que laboratórios remotos utilizados para educação online devem possuir alguns requisitos adicionais, devendo proporcionar ao usuário um ambiente onde estes possam compartilhar conhecimento, discutir e comparar resultados; enviar seus próprios experimentos para análise e discussão; emitir opiniões; enfim, participar do processo de desenvolvimento e atualização do laboratório. A seguir são apresentados dois laboratórios remotos utilizados para educação online.

2.5.1 Fieldbus Foundation Function Block Simulator (FBSIMU)

Em (MOSSIN, 2007), é proposto um laboratório remoto para ensino de sistemas de controle distribuído, utilizando o protocolo *fieldbus,* sendo este ambiente chamando *Fieldbus Foundation Function Block Simulator* (FBSIMU) Online (Figura 4). Este protocolo pode ser utilizado em sistemas de controle distribuído, possuindo uma arquitetura onde cada instrumento de campo possui inteligência própria e se comunicam através de barramento de dados. Através deste ambiente, alunos e professores da Escola de Engenharia de São Paulo, da Universidade de São Paulo, podem realizar simulações ou estudar o funcionamento do protocolo *fieldbus*. Além disso, o usuário poderá visualizar os fóruns disponíveis para discussão, salas de bate-papo, gerenciamento de arquivos e a lista de cursos disponíveis.

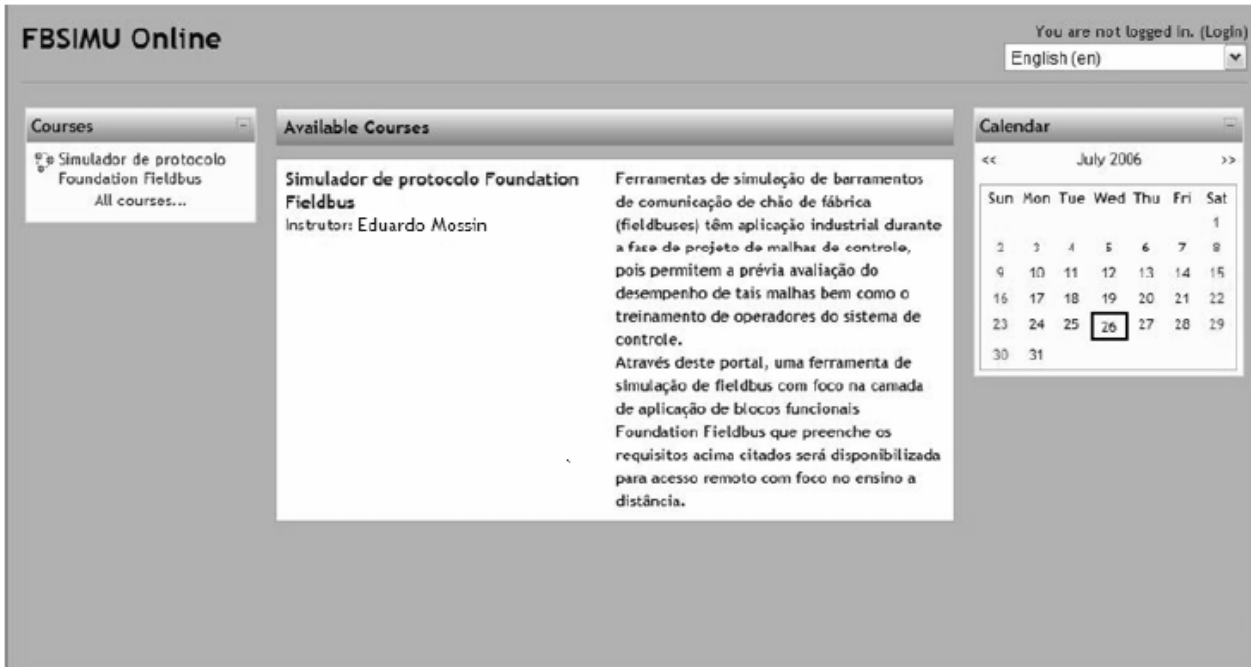

Figura 4 Ambiente FBSIMU Online
Fonte: (MOSSIN, 2007)

2.5.2 eMersion

Em (NGUYEN et al., 2005) é apresentado o ambiente eMersion (EMERSION, 2009), que foi desenvolvido na *École Polytechnique Federale de Lausanne (EPFL)*, na Suíça, com o objetivo de oferecer suporte para execução de experimentos remotamente dos dispositivos do

laboratório de física e/ou ferramentas para simulação computacional, sendo utilizado nos cursos de Automação e Controle e Fluidos Mecânicos oferecidos pela Escola de Engenharia da EPFL. Este ambiente possui dois componentes principais, o componente de experimentação, que permite a realização de experimentos, e o *eJournal*, para o compartilhamento e coleta de dados experimentais das atividades executadas por usuários (Figura 5).

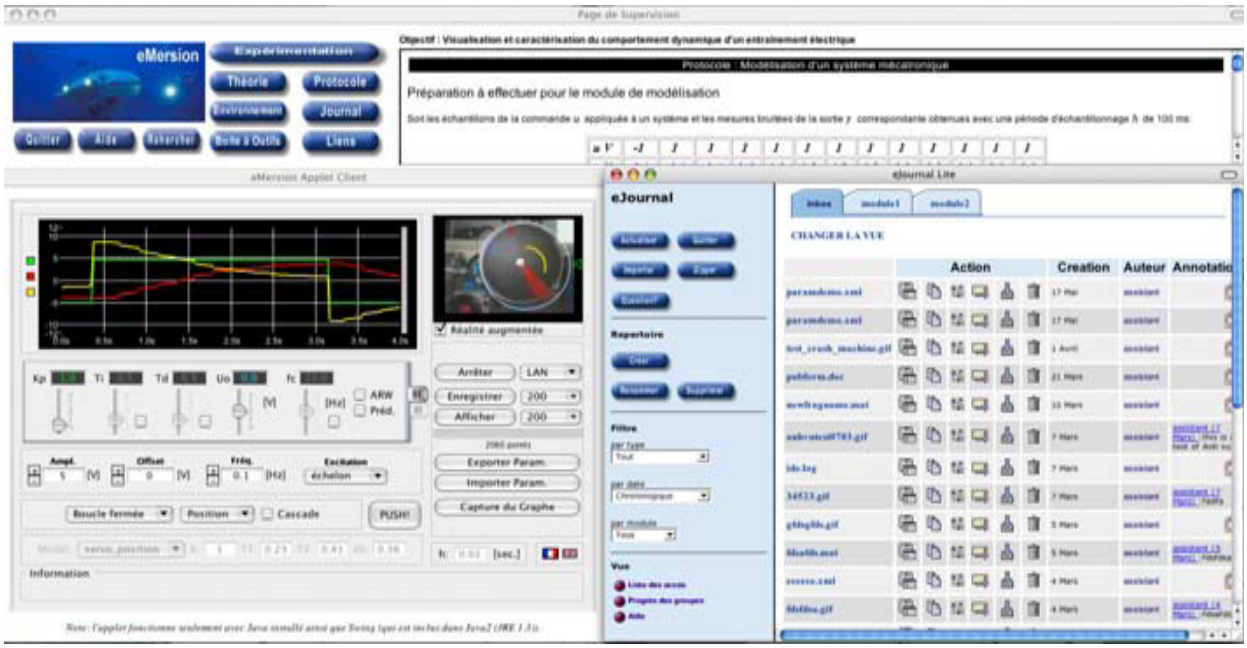

Figura 5 Ambiente eMersion
Fonte: (NGUYEN et al., 2005)

3 ANÁLISE DAS TECNOLOGIAS DISPONÍVEIS

Antes de definir a arquitetura utilizada no LabExp, constatou-se que distintas tecnologias poderiam atender os requisitos de um laboratório de experimentação remota, independente de sua finalidade. Sendo assim, neste capítulo pretende-se analisar estas tecnologias. No capítulo 4 é detalhada a arquitetura utilizada na implementação do Laboratório de Experimentação Remota em Tempo Real (LabExp). A seção 3.1 apresenta de modo conciso a internet e as redes de computadores, não se pretendendo esgotar uma discussão sobre este assunto, abordando, entretanto, alguns tópicos considerados mais relevantes no contexto deste trabalho. Na seção 3.2 conceitos, fundamentos, responsabilidades e exemplos de sistemas operacionais em tempo real são apresentados, pois experimentos

remotos de sistemas de controle necessitam obedecer a limites de tempo restritos, ou seja, funcionar em tempo real (DOZIO; MANTEGAZZA, 2003). A seção 3.3 apresenta ferramentas computacionais utilizadas para o desenvolvimento de sistemas de controle (*Computer Aided Control System Design Software* – CACSDS), caracterizando outra etapa fundamental. Finalmente, na seção 3.4 é apresentado o entendimento sobre laboratório de experimentação remota, assim como exemplos de laboratórios atualmente em funcionamento.

3.1 INTERNET E REDES DE COMPUTADORES

O modelo de um único computador atendendo a todas as necessidades computacionais de uma organização foi substituído pelas chamadas redes de computadores, nas quais os trabalhos são realizados por um grande número de computadores separados, mas interconectados. A fusão dos computadores e das comunicações teve uma profunda influência na forma como os sistemas computacionais eram organizados. A partir da década de 1990, com a evolução da internet se apresentou um novo paradigma, demonstrando o grande interesse de usuários pelo acesso gratuito a informação e pela comunicação escrita instantânea (KUROSE; ROSS, 2006). A internet pode ser utilizada comercialmente, residencialmente, e também voltada para a educação. Dizer que a internet pode ser utilizada para educação é bastante superficial, pois simplesmente o acesso gratuito a informação, disponível em diversos web sites, inclusive existindo grandes enciclopédias, como a Wikipedia (WIKIPEDIA, 2009), pode ser considerado uma fonte de conhecimento, ou seja, de educação. Entretanto aborda-se a internet aqui como meio para educação online.

3.1.1 Arquitetura de Rede

A maioria das redes é organizada como uma pilha de camadas ou níveis, colocadas umas sobre as outras. O número de camadas, o nome, o conteúdo e a função de cada camada diferem de uma rede para outra. No entanto, em todas as redes o objetivo de cada camada é oferecer determinados serviços às camadas superiores, isolando essas camadas dos detalhes

de implementação destes recursos. Em certo sentido, cada camada é uma espécie de máquina virtual, oferecendo determinados serviços à camada situada acima dela (KUROSE; ROSS, 2006).

Tratando-se desta organização em camadas, os conceitos de serviço, interface e protocolo são fundamentais. Um serviço é executado por uma camada para a camada acima dela, informando o que a camada faz, mas não infomando como as entidades acima dela o acessam ou como a camada funciona. A interface de uma camada informa como os processos acima dela podem acessá-la, especificando quais parâmetros e resultados são esperados, não revelando o funcionamento interno da camada. Os protocolos de uma camada fornecem os serviços oferecidos por essa camada, sendo as regras de comunicação das camadas (TANEMBAUM, 2003).

Uma arquitetura de camadas nos permite discutir uma parcela específica e bem definida de um sistema grande e complexo. Esta simplificação tem considerável valor intrínseco, pois provê modularidade fazendo com que a implementação de um serviço prestado pela camada torne-se mais fácil, desde que a camada forneça o mesmo serviço para a camada acima dela e utilize os mesmos serviços da camada abaixo dela, permanecendo o restante do sistema inalterado quando sua implementação é modificada (KUROSE; ROSS 2006).

Um conjunto de camadas e protocolos é chamado arquitetura de rede. A especificação de uma arquitetura deve conter informações suficientes para permitir que um implementador desenvolva o programa ou construa o hardware de cada camada, de forma que ela obedeça corretamente ao protocolo adequado. Entre as arquiteturas de redes, duas merecem atenção especial, os modelos *Open System Interconnection* (OSI) e *Transfer Control Protocol/Internet Protocol* (TCP/IP). O modelo OSI se baseia em uma proposta desenvolvida pela *International Standards Organization* (ISO) como a primeira etapa em direção à padronização internacional dos protocolos empregados nas diversas camadas. O modelo OSI possui sete camadas: aplicação, apresentação, sessão, transporte, rede, enlace e física. Embora os protocolos associados ao modelo OSI raramente sejam usados nos dias de hoje, o modelo em si é de fato bastante geral e ainda válido, e as características descritas em cada camada ainda são muito importantes (TANEMBAUM, 2003). Entretanto será interessante apresentar com maiores detalhes o modelo TCP/IP, pois este é utilizado na maior de todas as redes de computadores geograficamente distribuídos, a internet, e nesta dissertação.

3.1.1.1 Modelo TCP/IP

O modelo TCP/IP recebe este nome devido a seus dois principais protocolos, o *Transmission Control Protocol* (TCP) e o *Internet Protocol* (IP). Na figura 6 observa-se a estrutura de camadas deste modelo, observando que cada camada atende ou responde as solicitações da camada imediatamente ligada a ela.

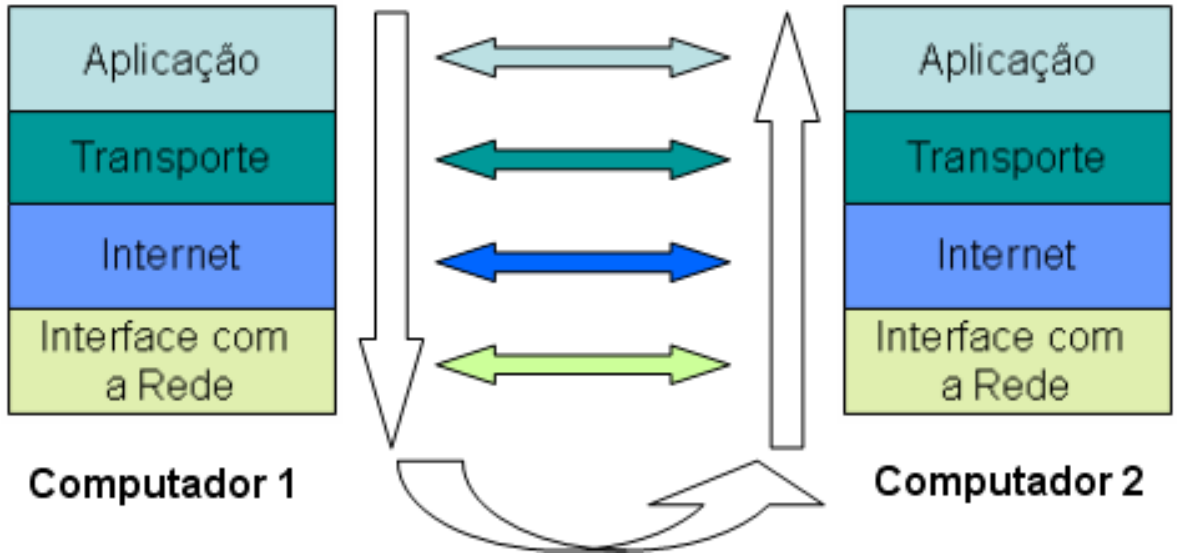

Figura 6 Arquitetura do Modelo TCP/IP

Na camada de Aplicação encontram-se os protocolos de aplicação, tais como o *Simple Mail Transfer Protocol* (SMTP), para envio e recebimento de e-mails, o *File Transfer Protocol* (FTP), para a transferência de arquivos e o *HyperText Transfer Protocol* (HTTP), para navegação web. Cada tipo de programa se comunica com um protocolo de aplicação diferente, dependendo da finalidade do programa. A camada de aplicação comunica-se com a camada de transporte através de uma porta, sendo estas portas numeradas. Algumas aplicações, consideradas padrão, utilizam o mesmo número de porta para comunicar-se com a camada de transporte. O uso de um número de porta permite ao protocolo de transporte saber qual o tipo de conteúdo do pacote de dados e, no receptor, saber para qual protocolo de aplicação ele deverá entregar o pacote de dados (TORRES; LIMA, 2007).

Após processar a requisição do programa, o protocolo na camada de Aplicação se comunicará com outro protocolo, na camada de Transporte, normalmente o TCP ou o *User Datagram Protocol* (UDP). O TCP é um protocolo orientado a conexões confiável que

garante que os dados enviados pelo processo remetente cheguem corretamente e em ordem ao processo destinatário. Esta transmissão segura é garantida através da utilização de controle de fluxo, números de seqüência, reconhecimentos e temporizadores. Esse protocolo fragmenta o fluxo de bytes de entrada em mensagens discretas e passa cada uma delas para a camada imediatamente inferior, a camada Internet. O UDP é um protocolo não orientado a conexão, portanto não garante que os dados enviados por um processo cheguem intactos ao destinatário. Entretanto, por não realizar todas as tarefas realizas pelo TCP, que visa garantir a entrega segura dos dados, estes podem ser entregues mais rapidamente, caracterizando importância para aplicações que necessitem de rapidez, como na utilização de dispositivos multimídia através da internet, como vídeo e áudio (KUROSE; ROSS, 2006).

A camada Internet garante que dados enviados por um remetente cheguem ao destinatário. Esta camada possui duas funções importantes: repasse e roteamento. Repasse refere-se à ação local realizada por um roteador para transferir um pacote da interface de um enlace de entrada para uma interface de um enlace de saída. Roteamento refere-se ao processo de âmbito geral da rede que determina os caminhos fim-a-fim que os pacotes devem percorrer desde o remetente até o destinatário, existindo diversos algoritmos de roteamento para essa finalidade (KUROSE; ROSS, 2006). Cada roteador possui uma tabela de repasse, onde esta tabela indica para qual das interfaces de enlace do roteador o pacote deve ser repassado, baseado no valor de um campo no cabeçalho. Os valores das tabelas de repasse são definidos por algoritmos de roteamento. Na figura 7 é mostrado como acontece o repasse e roteamento.

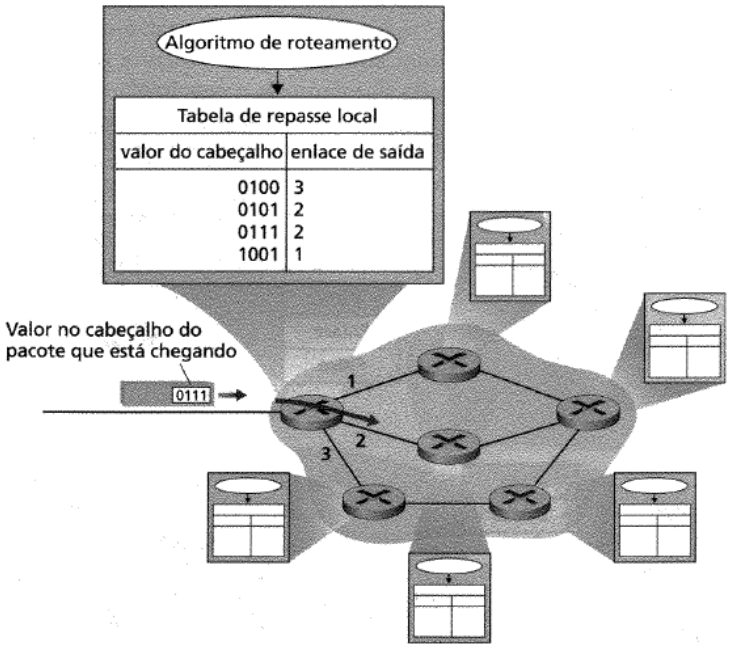

Figura 7 Repasse e Roteamento
Fonte: (KUROSE; ROSS, 2006)

Para realizar estas funções, na camada Internet encontra-se o IP, que recebe os pacotes enviados pela camada de Transporte e adiciona informações de endereçamento virtual, isto é, adiciona o endereço do computador que está enviando os dados e o endereço do computador que receberá os dados. Esses endereços virtuais são chamados endereços IP. Em seguida os pacotes são enviados para a camada imediatamente inferior, a camada Interface com a Rede.

A camada Interface com a Rede receberá os pacotes enviados pela camada Internet e os enviará para a rede. O que está dentro desta camada dependerá do tipo de rede que o computador estiver utilizando. Atualmente praticamente todos os computadores utilizam um tipo de rede chamado Ethernet. A Ethernet é dividida em três camadas: *Logical Link Control* (LLC), *Media Access Control* (MAC) e Física.

A LLC é definida pelo padrão IEEE 802.2. Este protocolo oculta a diferença entre diversos tipos de rede 802, fornecendo um único formato e uma única interface com a camada superior (TANEMBAUM, 2003). É responsável por adicionar informações de que protocolo na camada Internet foi o responsável por gerar os dados. Dessa forma, durante a recepção de dados da rede esta camada no computador receptor tem que saber que protocolo da camada de Internet ele deve entregar os dados.

A camada MAC é responsável por montar o quadro que será enviado para a rede. Esta camada é responsável por adicionar o endereço MAC de origem e de destino. Os quadros que são destinados a outras redes utilizarão o endereço MAC do roteador da rede como endereço de destino. Esta camada é definida pelo padrão IEEE 802.3, se uma rede com cabos estiver sendo usada, ou pelo padrão IEEE 802.11, se uma rede sem fio estiver sendo usada. A camada Física é a responsável por converter o quadro gerado pela camada MAC em sinais elétricos (se for uma rede cabeada) ou eletromagnéticos (se for uma rede sem fio) (TORRES; LIMA, 2007).

As camadas LLC e MAC adicionam suas informações de cabeçalho ao datagrama recebido da camada Internet. Uma estrutura completa de quadros gerados por essas duas camadas pode ser vista na figura 8. Note que os cabeçalhos adicionados pelas camadas superiores são vistos como “dados” pela camada LLC. A mesma coisa acontece com o cabeçalho inserido pela camada LLC, que será visto como dado pela camada MAC (TORRES; LIMA, 2007).

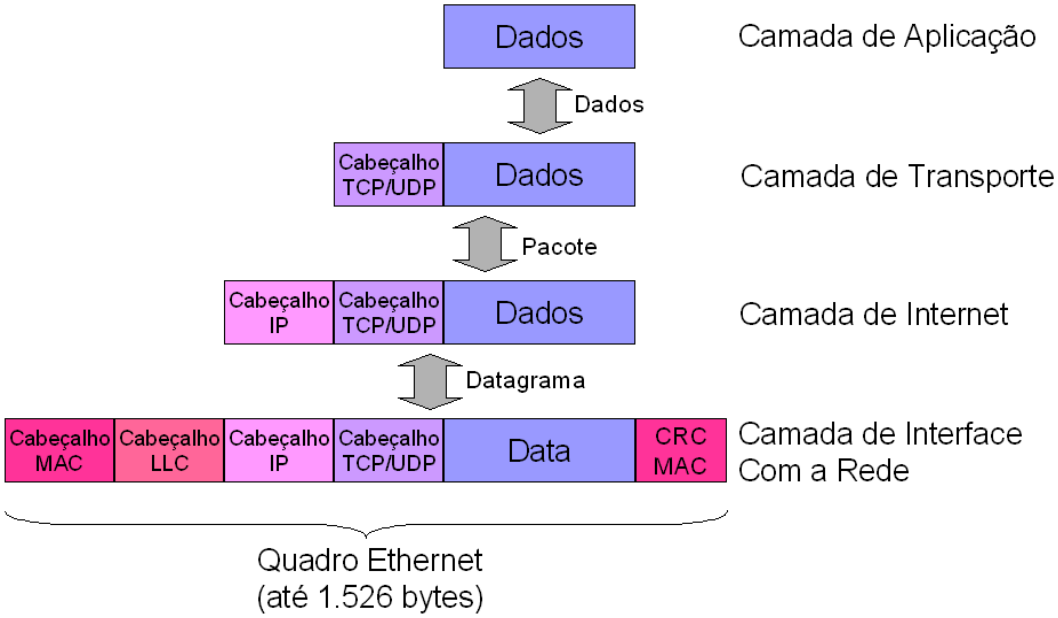

Figura 8 Estrutura completa de quadros
Fonte: (TORRES; LIMA, 2007)

3.2 SISTEMAS OPERACIONAIS EM TEMPO REAL

Atualmente, a quantidade de aplicações que necessitam de sistemas operacionais em tempo real para trocar dados, comandos e eventos através da utilização da Ethernet ou outros tipos de redes cresce consideravelmente. Essas aplicações variam muito em relação à complexidade e as necessidades de garantia no atendimento a restrições de tempo. Entre os sistemas mais simples, podem-se destacar controladores inteligentes embarcados, como sistemas embarcados para rastreamento de veículos. Por outro lado, referentes à complexidade estão sistemas de controle industriais, por exemplo. Além disso, alguns sistemas de tempo real apresentam restrições de tempo mais rigorosas que outros. Entre aplicações críticas estão sistemas de supervisão e controle de plantas industriais e sistemas embarcados em robôs, e entre as que apresentam restrições não tão críticas estão às teleconferências através da internet e aplicações de multimídia em geral (FARINES et al., 2000). Segundo (ABBOT, 2003):

"Tempo real não significa responder a eventos em velocidade real, mas responder em velocidade confiantemente suficiente, em prazos específicos, no contexto em que o sistema está operando."

Conseqüentemente, em cada resposta, o sistema de tempo real deve entregar um resultado correto dentro de um prazo determinado, ou poderá ocorrer uma falha temporal. Assim, o comportamento de um sistema de tempo real não depende somente da integridade dos resultados obtidos, mas também dos valores de tempo em que são produzidos. Uma reação que ocorra além do prazo determinado poderá ser inútil ou causar uma falha fatal no sistema.

Portanto, no contexto da tecnologia da computação, um sistema em tempo real deve ser capaz de executar tarefas onde um limite de tempo é o fator mais importante. Deve possuir comportamento determinístico, deve ser rápido e previsível, onde rapidez significa responder a eventos externos assíncronos em pouco tempo, e previsível significa prever o tempo final de execução de uma determinada tarefa (ABBOT, 2003).

Basicamente os sistemas em tempo real são qualificados em tempo real leve (*soft real time* - SRT) e tempo real rígido (*hard real time* - HRT). Em *s*istemas SRT o limite de tempo não é tão restrito, ou seja, a existência de atrasos não comprometeria toda a operação, como, por exemplo, na execução de aplicações multimídia, como vídeos, através da internet. Por sua vez, em sistemas HRT o limite de tempo é um fator fundamental, e atrasos podem comprometer toda a operação. Sistemas de controle industriais e sistema para controle de um servomotor, por exemplo, são considerados HRT, pois atrasos podem alterar o resultado do processo. Tendo em vista que o algoritmo de controle de sistemas HRT depende de um período de amostragem regular, havendo atraso nestas amostras, o algoritmo poderá tornar-se instável (DOZIO; MANTEGAZZA, 2003).

3.2.1 Responsabilidades de Sistemas Operacionais de Tempo Real

Tanto o sistema operacional Linux quanto o Microsoft Windows foram desenvolvidos para funcionar como sistemas de propósitos gerais. Em sistemas de propósitos gerais a latência (intervalo de tempo que um evento ocorre até o tempo em que o sistema responde aquele evento) não é tratada como prioridade. Ou seja, usuários comuns, que normalmente utilizam sistemas de propósitos gerais, não percebem se a latência for da ordem de 200 milisegundos ou 20 milisegundos. Entretanto, para sistemas em tempo real, esta latência deve ser precisa, havendo possibilidade de causar instabilidade no sistema em execução.

Os sistemas operacionais de propósitos gerais possuem diversas responsabilidades, entretanto, tratando-se de sistemas operacionais em tempo real, as seguintes destacam-se (BRUYNINCKX, 2002).

3.2.1.1 Escalonamento e gerenciamento de tarefas

Gerenciamento de tarefas é uma das principais funções do sistema operacional, pois tarefas devem ser criadas e apagadas enquanto o sistema estiver sendo executado. Tarefas podem mudar seus níveis de prioridade, os seus limites de tempo e as suas necessidades de memória, por exemplo. Em sistemas operacionais de tempo real o gerenciamento de tarefas deve ser mais cuidadoso, comparando-se com sistemas operacionais de propósitos gerais, se uma tarefa em tempo real é criada, ela deve utilizar a memória que necessita, sem atraso, e esta memória deve permanecer bloqueada na memória principal, a fim de evitar latências devido à ocorrência de *swapping* (remoção de processos da memória) (ABBOT, 2003). A alteração das prioridades em tempo de execução influencia o comportamento de todo o sistema e, conseqüentemente, a previsibilidade, que é tão importante para um sistema operacional em tempo real.

Múltiplas tarefas permanecem em execução em um sistema operacional, como administrar os recursos (processador e memória, por exemplo) que estas tarefas compartilham é uma tarefa fundamental em um sistema de tempo real. O processador é um dos recursos mais importantes, e a decisão de como compartilhá-lo é conhecida como escalonamento. A concessão geral de algoritmos de escalonamento está entre simplicidade (e eficiência) e otimalidade do algoritmo. Algoritmos que são ótimos geralmente são complexos ou requerem conhecimento sobre diversos parâmetros de tarefas, que não são facilmente encontrados. Sistemas operacionais de tempo real utilizam algoritmos de escalonamento simples, pois estes demandam pouco e determinístico tempo de computação (BRUYNINCKX, 2002).

Sistemas operacionais de propósitos gerais e de tempo real diferem consideravelmente quanto ao algoritmo de escalonamento utilizado. Ambos utilizam os mesmos princípios, mas são aplicados diferentemente, pois devem satisfazer diferentes critérios de desempenho. Um sistema de propósito geral objetiva máxima média de taxa de transferência e sistemas de tempo real objetivam comportamento determinístico.

3.2.1.2 Tratamento de interrupções

Um sistema operacional não deve somente ser capaz de escalonar tarefas de acordo com um determinado algoritmo, deve também servir os periféricos de hardware, como temporizadores, motores, sensores, dispositivos de comunicação, discos, entre outros. Todos estes podem requerer atenção do sistema operacional de modo assíncrono, isto é, no momento em que quiserem utilizar serviços do sistema operacional, o sistema deverá estar pronto para atender a requisição. Esta requisição é conhecida como interrupção.

Um sistema operacional de tempo real deve estar pronto para atender as interrupções imediatamente, sem atrasar qualquer processo de tempo real em execução (FARINES et al. 2000).

3.2.1.3 Comunicação inter processo

A comunicação inter processo abrange uma série de prerrogativas de programação que o sistema operacional disponibiliza para tarefas que necessitem realizar troca de informação com outras tarefas, ou sincronizar suas operações. Um sistema operacional em tempo real deve garantir que esta comunicação e sincronização funcionem de modo determinístico, sendo assim, deve permitir que as tarefas sejam executadas sem haver preocupação com os detalhes de sua implementação e as dependências de hardware (BRUYNINCKX, 2002).

3.2.1.4 Gerenciamento de memória

Diferentes tarefas do sistema operacional necessitam de parte da memória disponível. Assim, o sistema operacional deve disponibilizar a cada tarefa a memória necessitada (alocação de memória), mapear a memória real dentro dos limites de endereço utilizados por diferentes tarefas (mapeamento de memória) e realizar a ação apropriada quando uma tarefa utiliza memória que não foi alocada.

Em sistemas de tempo real, o gerenciamento desta memória deverá ser determinístico, priorizando sempre tarefas de tempo real, visando sempre haver disponibilidade de memória para estas tarefas (FARINES et al., 2000).

3.2.2 Exemplos de Sistemas Operacionais de Tempo Real

Conforme especificado anteriormente, tanto o sistema operacional Linux quanto o Microsoft Windows foram desenvolvidos como sistemas operacionais de propósitos gerais. Entretanto o Linux possui distribuições gratuitas disponíveis com comportamento de tempo real. Além das distribuições gratuitas, destaca-se a existência de distribuições comerciais, como o RTLinux/Pro da FSMLabs (RTLINUXPRO, 2009), o uLinux da Lineo Solutions (ULINUX, 2009) e o LynxOS da LynuxWorks (LYNXOS, 2009). Algumas distribuições gratuitas são destacadas a seguir.

3.2.2.1 RTLinux (Real Time Linux)

Em 1996, um grupo comandado por Victor Yodaiken na NMT (New Mexico Institute of Mining and Technology), apresentou um sistema Linux em tempo real, o RTLinux, provendo informações sobre o kernel do Linux, o hardware e as modificações necessárias para dispor de escalonamento determinístico e preemptivo. Em 1999 a companhia FSMLabs foi fundada e a partir de então dois produtos distintos começaram a existir, o RTLinux/Pro e o RTLinux/Free (FRANKE, 2009). O RTLinux é um pequeno kernel HRT que designa ao kernel padrão do Linux as tarefas de menor prioridade. Para isso, algumas linhas de código são inseridas no kernel padrão do Linux, como um patch, e a recompilação gera o RTLinux. No processo de recompilação, as configurações de interrupção, que são fundamentais no processo de habilitar e desabilitar interrupções, são substituídas. Assim, o RTLinux controla todas as interrupções (SANKARAYOGI, 2005). Para melhor entender o RTLinux, pode-se visualizá-lo como uma camada entre o kernel padrão do Linux e os dispositivos de hardware, sendo este processo realizado através da inicialização de módulos do RTLinux. O RTLinux

também substitui os escalonadores do Linux padrão. O kernel de tempo real do RTLinux é visto como hardware pelo kernel padrão do Linux e então intercepta todas as interrupções de hardware para atingir a performance de tempo real (SANKARAYOGI, 2005).

Em geral, qualquer modificação em um sistema operacional visando suportar aspectos de tempo real e não tempo real pode complicar este sistema operacional e resultar em um sistema operacional não confiável e ineficiente. Para evitar isto, o RTLinux separa as partes de tempo real e não tempo real do sistema operacional. Assim, a mínima latência de resposta a uma interrupção pode ser alcançada independentemente das atividades normais do kernel do Linux.

3.2.2.2. Xenomai

Em 2001 a primeira versão do Xenoadaptor, renomeado em seguida para Xenomai (XENOMAI, 2009), foi liberada, como um esforço independente para criar uma estrutura de tempo real para facilitar a migração de sistemas operacionais de tempo real para Linux.Em 2003, o projeto Xenomai foi fundido com o projeto RTAI gerando o projeto RTAI/Fusion. Seu maior objetivo foi alcançar capacidade de tempo real rígido para aplicações em nível de usuário comum, além de super usuário. Em 2005, o projeto RTAI/Fusion separou-se. O Xenomai reiniciou sua existência independente novamente, seguindo o design de um núcleo central abstrato. Seu principal objetivo é prover capacidade de tempo real baseada em algumas funcionalidades fundamentais exportadas por um sistema operacional de tempo real abstrato. Além disto, pode disponibilizar mecanismos para suportar aplicações que são originalmente desenvolvidas por sistemas de tempo real tradicional (FRANKE, 2009).

O Xenomai implementa um conjunto de serviços genéricos que são comuns para a maioria dos sistemas operacionais de tempo real. Estes serviços podem ser acessados pela Interface de Programação de Aplicativos (*Application Programming Interface* - API) nativa do Xenomai ou através de diferentes APIs para sistemas de tempo real, que são disponibilizadas.

3.2.2.3 RTAI (Real Time Application Interface)

Em 2000, com a apresentação do kernel 2.2.x e com a experiência desenvolvida por membros do DIAPM (Department of Aerospace Engineering, Politecnico di Milano), que estavam trabalhando em seu próprio RTLinux, resultaram na Real Time Application Interface (RTAI), visando a utilização do Linux para aplicações de tempo real. Por problemas de licença o projeto RTAI moveu sua camada abstrata de hardware (*hardware abstracion layer - HAL*) do RTLinux. A motivação para o desenvolvimento do RTAI iniciou com o reconhecimento das deficiências do RTLinux. A arquitetura do RTAI é semelhante a do RTLinux, a respeito de executar o kernel padrão do Linux como uma tarefa de prioridade mínima. A RTAI também utiliza a característica de inicialização de módulos, sendo orientado a módulos (ABBOT, 2003). A figura 9 apresenta a ordem cronológica em que estes sistemas foram desenvolvidos.

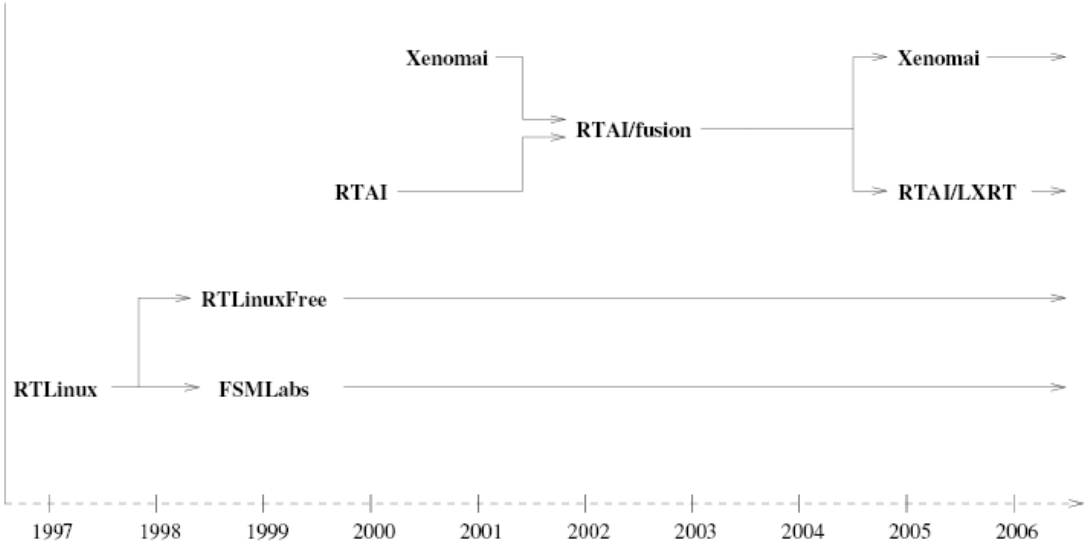

Figura 9 Ordem Cronológica RTLinux/Xenomai/RTAI
Fonte: (FRANKE, 2009)

Aparentemente o projeto RTLinux/Free foi abandonado, em detrimento ao RTLinux/Pro, e alguns desenvolvedores migraram para os projetos RTAI ou Xenomai. A RTAI é bastante semelhante ao Xenomai, como conseqüência de sua fusão destes projetos no passado. A RTAI não oferece a variedade de APIs para seus usuários, como o Xenomai, conseqüentemente não possuindo diferentes camadas de abstração acima uma das outras

(FRANKE, 2009). Entretanto a RTAI possui o RTAI-Lab, permitindo que usuários convertam diagrama de blocos de sistemas em executáveis RTAI, possibilitando a monitoração de sua operação. Os diagramas podem ser desenvolvidos usando tanto o gratuito Scilab/Scicos (SCILAB, 2009) quanto o comercial Matlab/Simulink/RTW (MATLAB, 2009).

3.3 CACDS (Computer Aided Control System Design Software)

Ferramentas computacionais para desenvolvimento de sistemas de controle (*Computer Aided Control System Design Software* - CACSDS) disponibilizam um ambiente para programação, modelagem e simulação de sistemas de controle e permitem análise de desempenho. Um CACSDS provê um ambiente de programação poderoso e flexível, assim como uma interface gráfica para usuário (G*raphical User Interface* - GUI) ajudando-o a construir o modelo do sistema.

Dois exemplos de CACSDS disponíveis atualmente são Matlab/Simulink e Scilab/Scicos. O Matlab/Simulink é um produto comercial e o Scilab/Scicos possui código aberto. Ambos possuem suporte para simulações em tempo real. O Matlab/Simulink através do workshop de tempo real (*Real Time Workshop* - RTW) e o Scilab/Scicos através do projeto de código aberto RTAICodeGen.

3.3.1 Scilab/Scicos/RTAICodeGen

O Scilab é um software científico desenvolvido pelo *Institut National de Recherche En Informática Et Automatique* (INRIA) na *Ecole Nationale Dês Ponts Et Chaussees* (ENPC) desde 1990. Este software é uma linguagem de programação de alto nível orientada a números. A linguagem disponibiliza um ambiente de programação interpretado, sendo matrizes o tipo de dado principal. Assim como este software provê simples operações matriciais, como a multiplicação, também disponibiliza bibliotecas para operações de alto nível. O Scilab pode ser utilizado para processamentos de sinais, análise estatística,

processamento de imagens e simulação dinâmica de fluídos, por exemplo (CAMPBELL et al., 2006).

Mesmo não sendo tão avançado quanto o Matlab, o Scilab pode comparar-se a este em vários aspectos. O Scilab disponibiliza uma ferramenta para simulação conhecida como Scicos, que é similar ao Simulink (ferramenta do Matlab) e é utilizado para modelagem e simulação de sistemas dinâmicos. O Scilab/Scicos é suportado pela maioria dos sistemas operacionais, como Windows e Linux.

Após sua instalação no sistema, o Scilab/Scicos pode ser integrado a RTAI e os arquivos necessários para geração de código podem ser instalados como macros do Scilab. Depois do processo de configuração, usuários podem ter acesso a opção de geração de código da RTAI (RTAICodeGen) através da GUI do Scicos. Assim, um usuário poderá desenvolver o modelo de um determinado sistema e gerar um módulo para execução em tempo real. Este processo é realizado através da execução do gerador de código da RTAI, nos superblocos do Scicos, que contem o modelo dinâmico do sistema. A execução dos arquivos gerados inicia as tarefas de tempo real.

3.3.2 Matlab/Simulink/RTW

O Matlab, que é o acrônimo de Matrix Laboratory, é uma ferramenta poderosa e de fácil utilização para computação matemática, provendo a linguagem de programação Matlab, bibliotecas de funções matemáticas, ambiente de desenvolvimento, gráficos e interface para programação de aplicações para as linguagens C e Fortran (MATSUMOTO, 2004).

O Simulink é um software integrado ao Matlab utilizado para modelagem, simulação e análise de sistemas contínuos, discretos e híbridos. O Simulink possui uma GUI que permite a construção de novos modelos através de uma biblioteca de blocos. Além disto, possui suporte para blocos definidos por usuários com suas *system functions* (S-functions), escritas em Matlab, C/C++, Fortran e Ada.

O RTW prove um ambiente para geração de código em C para prototipagem rápida e desenvolvimento. O RTW gera código fonte dos modelos do Simulink para criar aplicações de software de tempo real e é aplicável para qualquer modelo independente do seu modelo de tempo (contínuo ou discreto).

Para atingir um comportamento de tempo real rígido, o Matlab precisa ser combinado com um sistema operacional de tempo real, onde a aplicação desenvolvida será executada no espaço do kernel com capacidade de preempção e controle de interrupções. Conforme especificado na seção 3.2.2.3, a RTAI provê esta combinação.

Para integrar o Matlab/Simulink/RTW com a RTAI, faz-se necessário copiar para o diretório raiz do Matlab a RTAI-LIB, que é parte da RTAI e uma biblioteca consistindo de S-functions escrita em C. Esta biblioteca será compilada para gerar blocos de construção baseados na RTAI. A biblioteca criada provê interface para blocos Comedi (COMEDI, 2009), que permitem conectar-se com hardware externos e blocos para monitoração de sinais utilizando a GUI do RTAI-Lab disponibilizada.

Comedi é um projeto iniciado por David Schleef visando o desenvolvimento de drivers, ferramentas e bibliotecas para aquisição de dados, com código aberto. A equipe Comedi desenvolveu dois pacotes com propósito de aquisição de dados: Comedi e ComediLib. O Comedi é uma coleção de drivers para dispositivos para uma grande variedade de placas de aquisição de dados e provê os drivers na forma de módulos que podem ser iniciados pelo kernel do Linux. Comedilib é uma biblioteca que provê uma interface para os drivers Comedi. Esta biblioteca possui algumas ferramentas úteis para os usuários como programas de calibração e programas que demonstram seu funcionamento. Uma vez que o pacote Comedi esteja instalado, pode-se instalar o pacote Comedilib e testar/calibrar a placa utilizando a ferramenta de calibração para verificar se o driver do dispositivo para este hardware está funcionando.

Após a instalação da RTAI-LIB, o sistema a ser simulado em tempo real pode ser modelado utilizando os blocos disponíveis do Simulink. O código será gerado em um determinado diretório, compilado e conectado utilizando os arquivos de criação (*makefiles*) do RTAI para gerar o arquivo executável que será executado em tempo real. Depois que o executável for iniciado, a RTAI-LIB GUI será utilizada para monitorar os sinais durante a simulação.

3.4 LABORATÓRIO DE EXPERIMENTAÇÃO REMOTA

Um laboratório de experimentação remota consiste num agrupamento de instrumentos de propósito geral, conectados a um conjunto de sistemas de computadores pessoais, conectados à internet (WISINTAMER, 1999), permitindo (SILVA, 2001):

- Interações com o mundo físico, garantindo que os resultados obtidos sejam os mesmos que se obteria localmente;
- Possibilidade de interações com simuladores;
- Que os usuários tenham acesso a recursos que não possuem localmente, possibilitando que o número de usuários seja ilimitado;
- Custo reduzido para a realização de experimentos reais;
- Que experimentos possam ser realizados a qualquer hora e de qualquer lugar, desde que se tenha acesso à internet*;*
- Possibilidade de utilização mais ampla das redes disponíveis nas universidades e escolas técnicas brasileiras;
- Que usuário ganhe sentimento prático, mesmo não estando no laboratório presencial para a realização de experimentos.

As tecnologias utilizadas para a implementação de um laboratório de experimentação remota, independente de sua finalidade, divergem de acordo com sua necessidade. Por exemplo, na etapa local, onde a experiência é desenvolvida e os dados são processados, podem-se utilizar diferentes linguagens de programação, assim como utilizar softwares que possuem este objetivo, como os comerciais Matlab/Simulink e LabView (LABVIEW, 2009), e o gratuito Scilab/Scicos; a disponibilização da experiência para acesso através da internet pode ser feito através de distintos servidores web, como os gratuitos Apache (APACHE, 2009) e Jetty (JETTY, 2009), ou o comercial Microsoft IIS (MICROSOFT IIS, 2009); e para o desenvolvimento de uma interface onde o usuário possa interagir com a experiência podem-se utilizar distintas linguagens de programação, como HTML (CAMARGOS; MENEZES, 2008), PHP (CAMARGOS; MENEZES, 2008) e JSP (GONÇALVES, 2007), assim como ferramentas disponibilizadas por softwares de desenvolvimento, como o LabView.

3.4.1 Base de Conceitos

Neste tópico descreve-se, quais conteúdos um usuário de um laboratório de sistemas de controle pode lidar, ou qual a base de conceitos que se relaciona com a teoria de controle, além dela própria (SILVA, 2001). A Figura 10 ilustra, senão todos os conteúdos, os que se consideram mais relevantes. Nela, as fontes de problemas cujas soluções envolvem o uso de teoria de controle são diversas, passando deste à construção de protótipos didáticos, até o controle de sistemas aeroespaciais. Uma vez caracterizado o problema, os recursos humanos responsáveis por sua solução necessitam, usualmente, de um modelo matemático e os "produtos" (soluções do problema) que devem ser gerados podem ser: um modelo de simulação, um algoritmo de identificação ou um controlador (ou algoritmo de controle). Para obtê-los, necessita-se trabalhar com recursos de software e hardware, além da teoria de controle. Este trabalho ocorre de modo cíclico, ou seja, o que se gera, seja em software ou hardware, antes de ser considerado um produto final acabado, sofre realimentações em relação aos recursos humanos responsáveis pela solução do problema, e muitas vezes com a própria fonte do problema afim de que se garanta um produto final bem acabado.

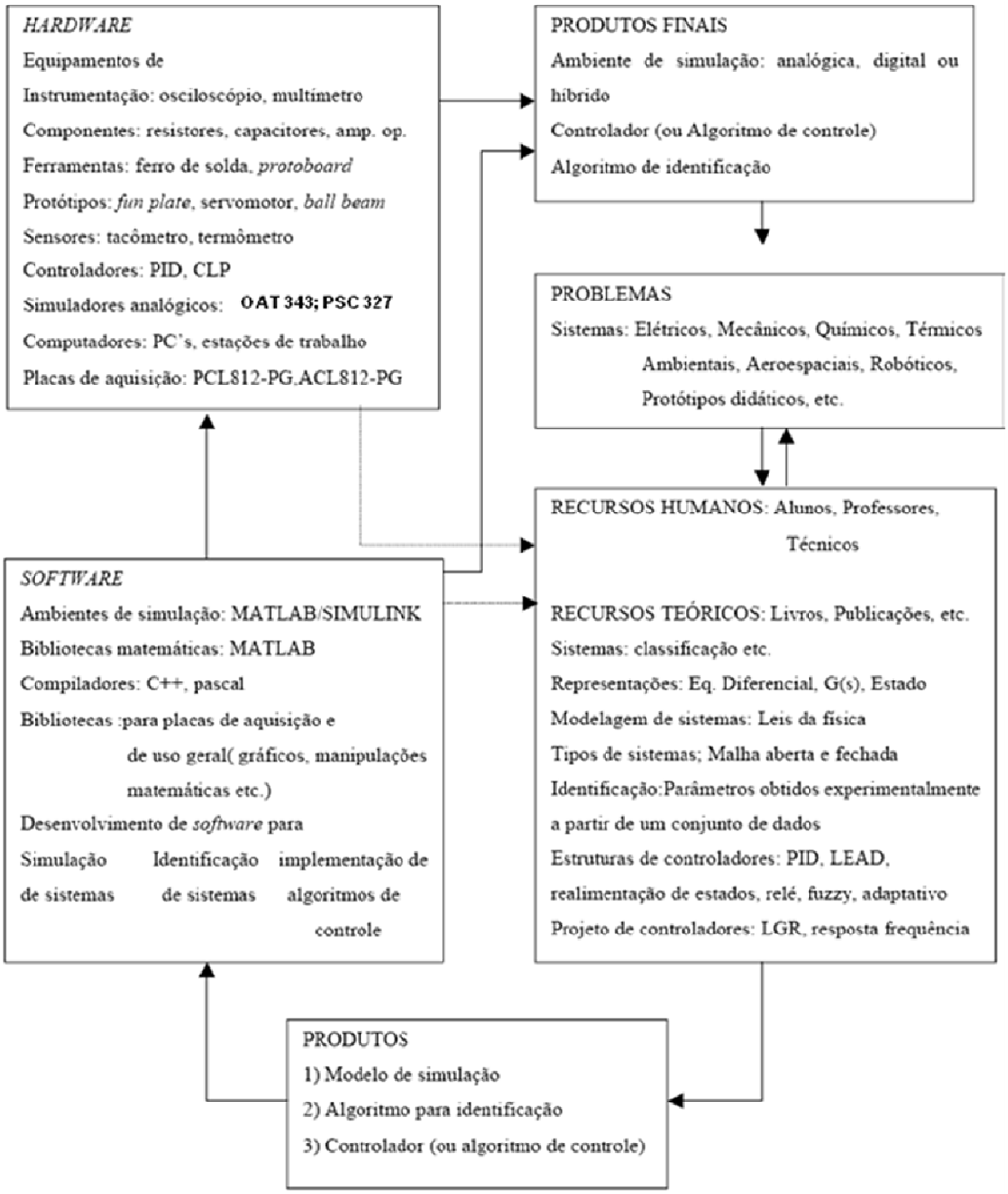

Figura 10 Base de Conceitos
Fonte: (SILVA, 2001).

A seguir alguns laboratórios de experimentação remota, que utilizam sistemas de controle, são apresentados.

3.4.2 MOCONET (Monitoring and Controlling Laboratory Processes over Internet)

A plataforma MOCONET (MOCONET, 2009) permite que usuários conduzam experimentos no laboratório remoto através da internet. A interface de comunicação com o usuário é um Java applet, portanto pode ser executada em qualquer web browser com permissão para Java. A figura 11 ilustra a interface da plataforma. Esta plataforma suporta a realização de testes, em processos reais, de controladores desenvolvidos utilizando o Matlab/Simulink (POHJOLA et al., 2006). Esta plataforma está em funcionamento e é utilizada em cursos de laboratório no Laboratório de Engenharia e Controle da Universidade de Tecnologia de Helsinki, Finlândia.

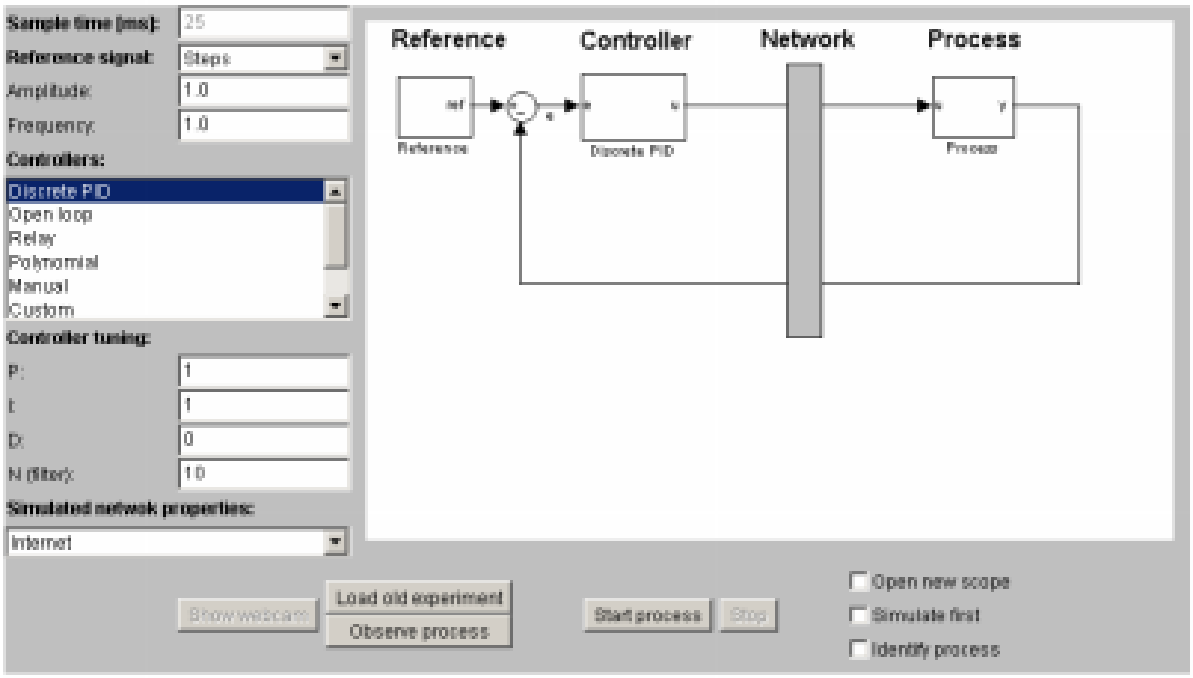

Figura 11 MOCONET
Fonte: (MOCONET, 2009)

3.4.3 ACT (Automatic Control Telelab)

O ACT (ACT, 2009) foi desenvolvido por um grupo do *Dipartimento di Ingegneria dell'Informazione* da Universidade de Siena, Itália. Neste laboratório, o usuário pode: desenvolver seu próprio controlador, utilizando Simulink, e testá-lo remotamente nos

experimentos disponibilizados pelo laboratório; controlar experimentos com os controladores pré-definidos ou utilizar seu próprio controlador; alterar parâmetros do experimento on-line, enquanto o experimento está em execução; e altear o sinal de referência do sistema (CASINI et al., 2004). O usuário pode realizar experiências reais de controle de posição e velocidade de um servomotor, controle de nível da água em um tanque, e controle de um sistema de levitação magnética (Figura 12).

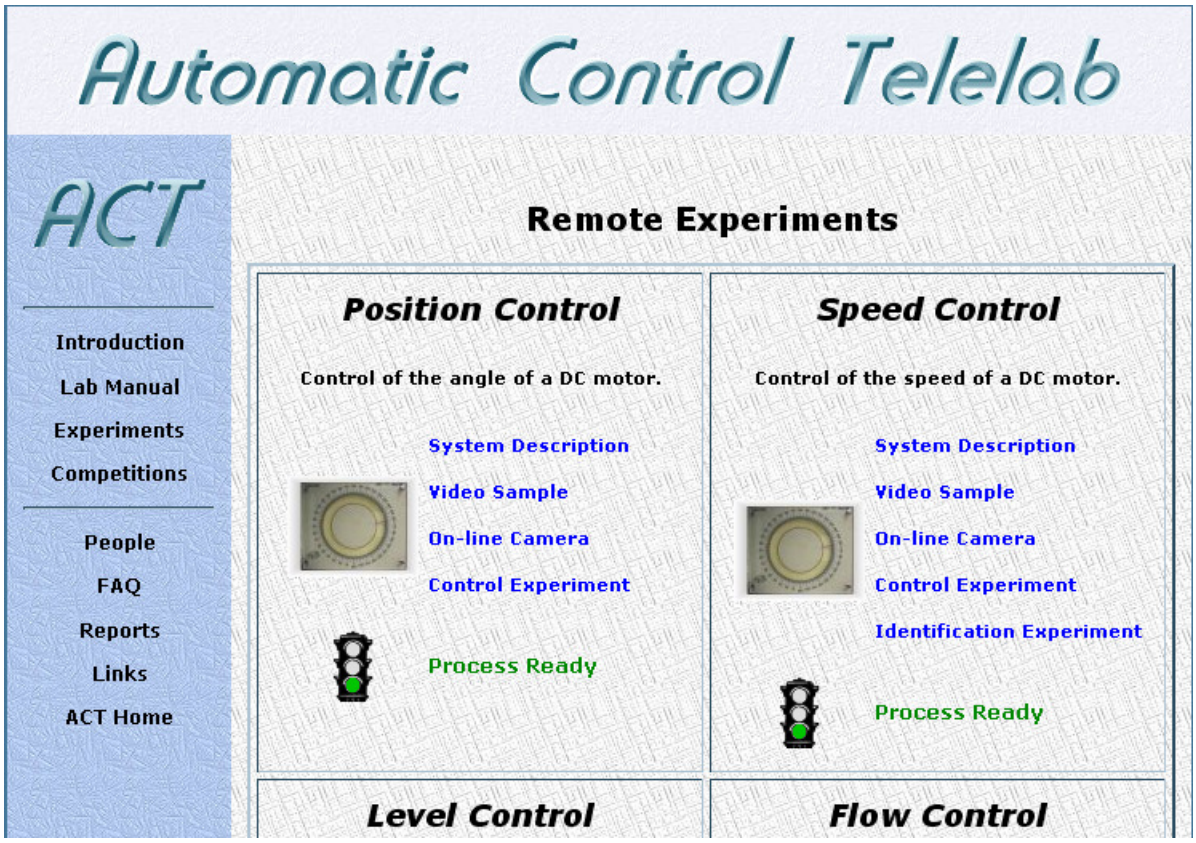

Figura 12 Automatic Control Telelab
Fonte: (ACT, 2009).

3.4.4 ARTIST (A Real-Time Interactive Simulink-based Telelab)

O ARTIST (ARTIST, 2009) foi implementado pelo *Automatic Control Group* do *Dipartimento di Sistemi e Informatica* da Universidade de Florença – Itália. Neste laboratório é permitido: que o controlador desenvolvido (arquivo simulink) seja automaticamente convertido como um executável em tempo real e seja carregado no computador remoto

conectado a um ou mais processos, garantindo performance em tempo real rígido; que experimentos sejam controlados por um navegador web padrão, permitindo modificação total dos parâmetros do controlador e observação dos sinais em gráficos gerados em tempo real; que, para cada experimento, um supervisor possa ser empregado, visando permitir que os usuários lidem com plantas instáveis e prevenindo que eles forcem o sistema com ações de controle equivocadas; e que experimentos possam ser facilmente compartilhados entre diferentes laboratórios ou entre diferentes universidades (BASSO; BAGNI, 2004). Neste laboratório pode-se realizar experiências reais de controle de um sistema de levitação magnética e posição de um helicóptero, e a simulação de um pêndulo invertido (Figura 13).

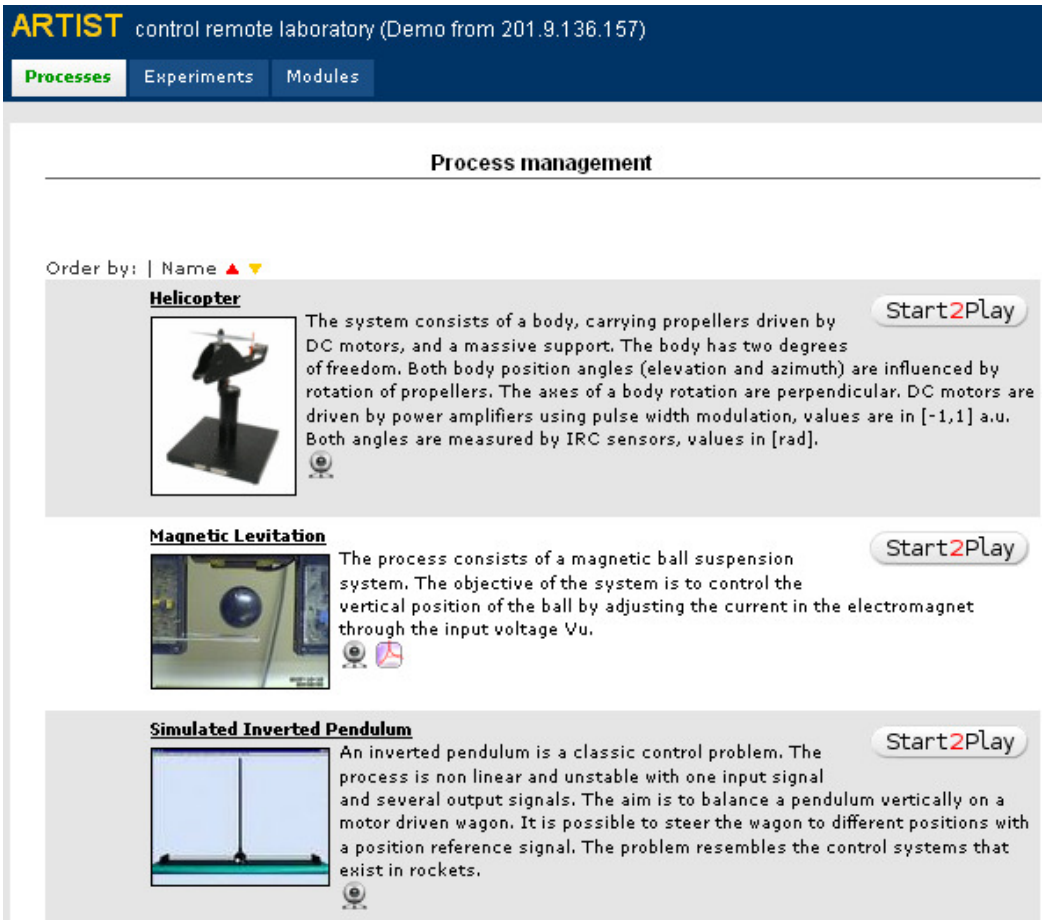

Figura 13 ARTIST
Fonte: (ARTIST, 2009)

4 ARQUITETURA DO LABORATÓRIO

Após a realização de considerações sobre aspectos relacionados à educação online e a análise das possíveis tecnologias que poderiam ser utilizadas no desenvolvimento de um laboratório de experimentação remota, foram definidas as tecnologias que seriam utilizadas no Laboratório de Experimentação Remota em Tempo Real (LabExp).

As tecnologias analisadas foram selecionadas de acordo com requisitos que permitissem o desenvolvimento de um ambiente onde os usuários, além de interagir com experimentos, pudessem interagir com outros alunos e participar do processo de desenvolvimento e manutenção do LabExp. Esta participação foi obtida através do desenvolvimento de aplicações que permitem: envio de experimentos e documentos; emissão de opiniões e resposta a questionário; e um fórum.

O LabExp proporciona acesso aos usuários através de páginas HTTP. Para isso o usuário necessitará apenas de um navegador web. O servidor web utilizado, responsável por atender as requisições HTTP, é o Apache (APACHE, 2009) devido ser gratuito, de fácil implementação e bastante utilizado.

O servidor web Apache é responsável por atender as requisições de páginas com conteúdo estático. As requisições de páginas com conteúdo dinâmico são atendidas por um servidor de aplicações. As aplicações disponíveis atualmente no laboratório foram desenvolvidas utilizando a linguagem de programação JSP. Esta linguagem necessita de um container para funcionar, por isso foi utilizado o Apache Tomcat (TOMCAT, 2009) como servidor de aplicações. Este container foi selecionado, pois apresenta fácil integração com o servidor web Apache, configuração simples, além de ser muito aceito entre a comunidade tecnológica.

Além do servidor web e de aplicações outro servidor foi utilizado: o servidor de experiências. Este servidor disponibiliza acesso remoto as experiências desenvolvidas localmente. Experimentos de sistemas de controle necessitam obedecer a limites de tempo restritos, ou seja, funcionar em tempo real (DOZIO; MANTEGAZZA, 2003), por isso foi utilizada a Interface de Aplicação de Tempo Real (*Real Time Application Interface* - RTAI). Para disponibilizar acesso remoto aos experimentos, garantindo o funcionamento dos mesmos em tempo real, foi utilizado o componente RTAI-XML, que implementa um servidor XML integrado com a RTAI (BASSO et al., 2005). A figura 14 representa a arquitetura do LabExp.

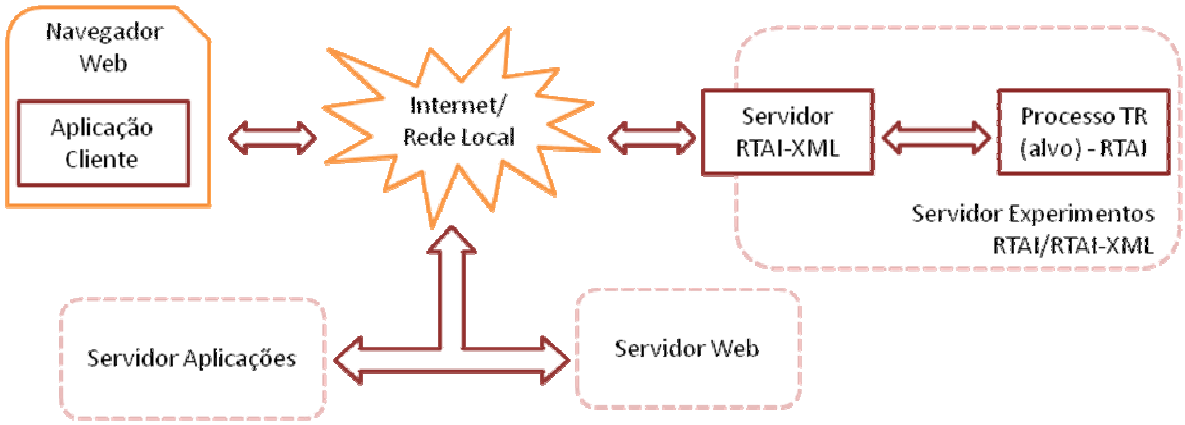

Figura 14 Arquitetura do LabExp

Neste capítulo serão apresentados maiores detalhes sobre os servidores web, de aplicações e de experiências.

4.1 SERVIDOR WEB

O servidor web Apache foi utilizado para atender as requisições de páginas HTTP estáticas por usuários, distribuídos tanto em redes locais, quanto através da internet. Este servidor possui código aberto, sendo seu código fonte disponibilizado de forma livre, de acordo com a *Apache Software License* (APACHE LICENSE, 2004). A figura 15 representa como o servidor web funciona no LabExp. O projeto do servidor web Apache é gerido conjuntamente por um grupo de voluntários, distribuídos mundialmente, utilizando a internet para se comunicar, planejar e desenvolver o servidor e sua documentação. Este projeto faz parte da Apache Software Foundation. Além disso, centenas de utilizadores contribuíram com idéias, códigos e documentação para o projeto.

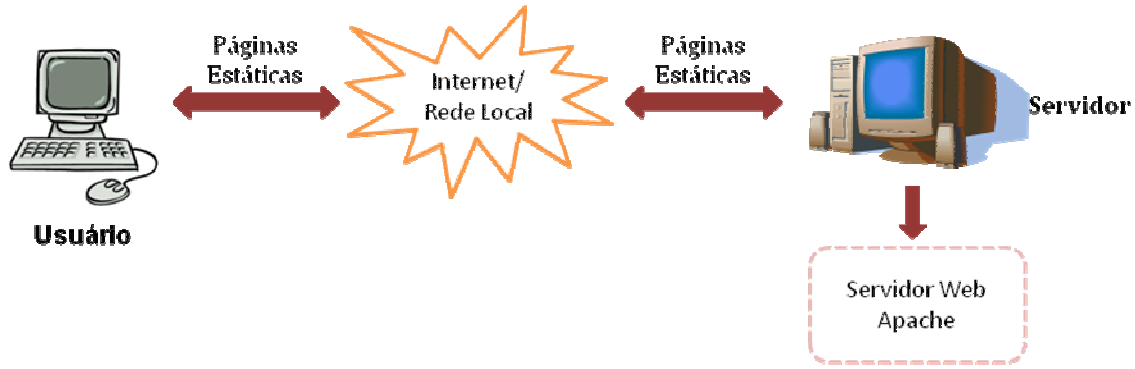

Figura 15 Arquitetura servidor Web Apache

O servidor Apache é o servidor web mais utilizado mundialmente (NETCRAFT, 2009), conforme se pode observar na figura 16. Esta pesquisa foi realizada de Dezembro de 2005 a Março de 2009. A Netcraft realiza pesquisas mensais sobre a utilização de servidores web em todo o mundo e em março de 2009 obteve a resposta de 224.749.695 sites. A grande utilização do Apache deve-se a sua praticidade, versatilidade e gratuidade (MARCELO, 2006).

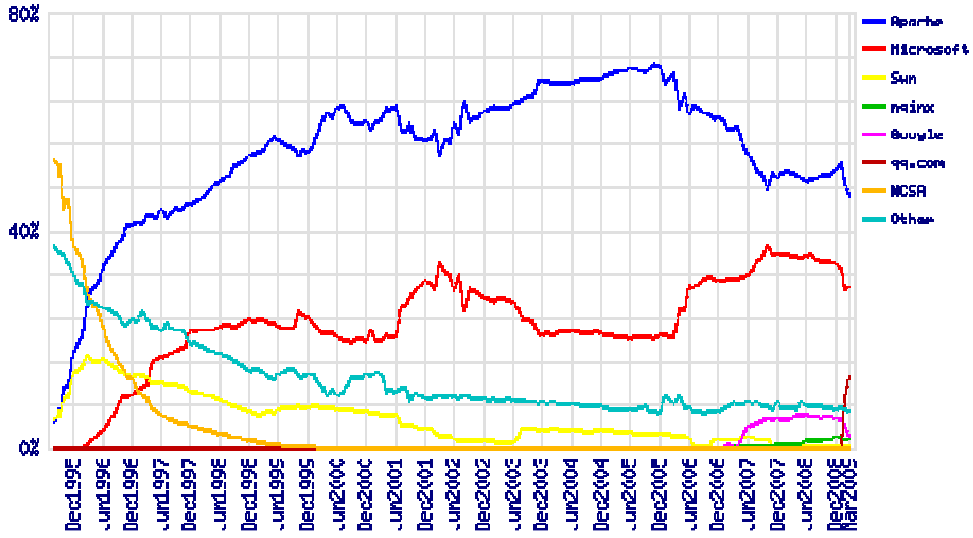

Figura 16 Compartilhamento do mercado pelos principais servidores web
Fonte: (NETCRAFT, 2009)

4.2 SERVIDOR DE APLICAÇÕES

O servidor de aplicações é responsável em processar as interfaces com conteúdo dinâmico. Existem três aplicações disponíveis no laboratório atualmente: a primeira permite ao usuário enviar experimentos e documentos, que descrevem os respectivos experimentos; a segunda envia as opiniões e permite aos usuários responder a questionários; e a terceira aplicação é um fórum implementado visando proporcionar aos usuários um mecanismo de interação, o JForum. O JForum (JFORUM, 2009) é um ambiente de discussão desenvolvido em Java, gratuito, que atendeu completamente as necessidades do LabExp.

A utilização destas aplicações permite que usuários compartilhem conhecimento e possam evoluir através de um aprendizado colaborativo, discutindo e comparando resultados com outros alunos, tutores e professores.

Estas três aplicações foram desenvolvidas utilizando a linguagem de programação JavaServer Page (JSP). A JSP precisa de um container para operar. O servidor de aplicações utilizado no LabExp é o Apache Tomcat, que é o container referência de implementação (RI) para JSP e servlets (MOODIE, 2007). Este servidor, além de atender as necessidades do LabExp, permiti emitir relatórios sobre os usuários que realizarem acesso, apresentando, por exemplo, horário e data de acesso.A figura 17 representa o funcionamento deste servidor de aplicações.

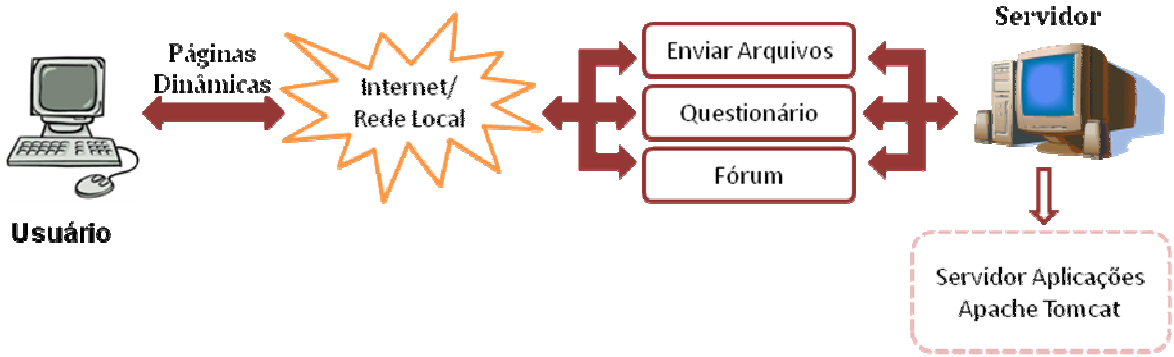

Figura 17 Arquitetura servidor de aplicações Apache Tomcat

A prioridade principal do Tomcat é ser totalmente concordante com as especificações dos Servlet e JSP publicadas pela Sun Microsystems. Uma RI possui o beneficio adicional de refinar estas especificações, independente da tecnologia utilizada. Como os desenvolvedores da Sun adicionam código por especificação, eles podem detectar problemas nos requerimentos de implementação e conflitos com a especificação, e como a RI é liberada com a especificação, o Tomcat sempre será o primeiro servidor a disponibilizar as novas características da especificação, quando finalizada (MOODIE, 2007).

O Tomcat é desenvolvido em ambiente aberto e participativo e liberado através da *Apache Software License*. O Apache Tomcat pretende ser uma colaboração dos melhores desenvolvedores de todo o mundo (TOMCAT, 2009).

4.3 SERVIDOR DE EXPERIÊNCIAS

O servidor de experiências é responsável por prover aos usuários comunicação remota com os experimentos desenvolvidos no servidor local. Experimentos remotos de sistemas de controle necessitam obedecer a limites de tempo restritos, ou seja, funcionar em tempo real

(DOZIO; MANTEGAZZA, 2003). Por isso foi utilizada a Real-Time Application Interface (RTAI), com o componente RTAI-XML, como servidor de experiências.

4.3.1 Real Time Application Interface (RTAI)

Conforme especificado anteriormente, o desenvolvimento de aplicações de sistemas de controle requer comportamento em tempo real. O termo tempo real pode possuir significados diferentes, dependendo da aplicação. Anteriormente também foi ratificado que basicamente os sistemas em tempo real são qualificados em tempo real leve (SRT) e tempo real rígido (HRT), lembrando que *e*m *s*istemas SRT o limite de tempo não é tão restrito, ou seja, a existência de atrasos não comprometeria toda a operação, como, por exemplo, na execução de vídeos, ou outras aplicações multimídia, através da internet. Por sua vez, em sistemas HRT o limite de tempo é um fator fundamental, e atrasos podem comprometer toda a operação. Um exemplo típico de sistemas HRT seria um sistema controlador (computador) e sistema controlado (planta), como sistemas de controle industriais e sistema para controle de um servomotor. A figura 18 representa o servidor RTAI atendendo a experimentos de tempo real.

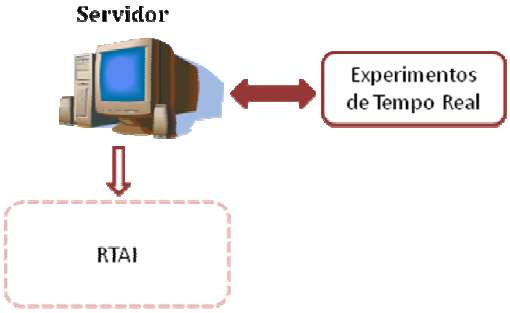

Figura 18 Arquitetura RTAI

A RTAI funciona como um *patch* para o sistema operacional Linux garantindo seu funcionamento em tempo real (DOZIO; MANTEGAZZA, 2003). A aplicação deste *patch* implicou na recompilação do kernel do Linux, informando parâmetros como configuração do processador, memória, etc. O Linux foi utilizado no LabExp pois, entre outros motivos, é gratuito, estável e, principalmente, pela possibilidade de manipulação do seu kernel, ratificando que este sistema foi desenvolvido para realização de tarefas de propósito geral, não possuindo características de tempo real (SILVA, 2007).

Esta interface utiliza o conceito de Hardware Abstraction Layer (HAL) modificando o kernel do Linux padrão, criando o RTHAL. Esta modificação consiste em uma estrutura de ponteiros para os vetores de interrupção e as funções de habilitar/desabilitar interrupções. Aproveitando-se da possibilidade de inicialização de módulos pelo kernel do Linux, a RTAI possui seus próprios módulos, onde, antes de sua inicialização, os ponteiros referem-se às funções do kernel do Linux padrão. Após a inicialização dos módulos, esses ponteiros mudam para trabalhar sob controle da RTAI. Como vantagem, apresenta-se pouco impacto no kernel do Linux, alterando-se ou acrescentando-se, aproximadamente, 70 linhas de código no kernel (DOZIO; MANTEGAZZA, 2003).

A RTHAL possui três funções principais (DOZIO; MANTEGAZZA, 2003):

- Concentrar todos os ponteiros para os dados e funções internos do kernel em uma única estrutura, para permitir que todas as funcionalidades do kernel que são importantes para aplicações de tempo real sejam facilmente tratadas, assim elas serão substituídas dinamicamente pela RTAI quando o comportamento de tempo real for necessário;
- Refazer as funções, estruturas de dados e macros relacionados ao Linux, criando a possibilidade de utilizá-los para inicializar os ponteiros da RTHAL para operações normais do Linux;
- Alterar o Linux para utilizar o que estiver direcionado para a RTHAL para suas operações.

Com os módulos centrais da RTAI instalados, o Linux poderá estender sua execução para o domínio de tempo real rígido. Assim, além de fornecer Linux totalmente preemptível, em nível de tratamento de interrupções, a RTAI provê serviços de escalonamento que são executados em tempo real e ferramentas de comunicação eficientes para permitir interação com as tarefas padrão do Linux.

O escalonador RTAI totalmente preemptível pode escalonar diretamente de dentro dos manipuladores de interrupção, assim o Linux não poderá atrasar qualquer atividade de tempo real da RTAI. Alguns serviços oferecidos por estes escalonadores, tanto em nível de usuário quanto kernel, são apresentados a seguir (DOZIO; MANTEGAZZA, 2003):

- Gerenciamento básico de tarefas: gerenciamento de tempo e conversões, designação de prioridade dinâmica, designação de políticas de escalonamento, bloqueio e desbloqueio de escalonadores, etc;

- Gerenciamento de memória: compartilhamento de memória inter processos, compartilhamento de dados em nível de usuário e kernel, alocação dinâmica de memória;
- Semáforos: esperar, enviar, espalhar em contagem, binário e recursos com heranças de prioridade máxima, visando evitar inversão de prioridade;
- Caixas de mensagem: enviar e receber mensagens com capacidade de multi leitura e escrita, mensagem em fila tanto em ordem FIFO quanto prioridade. Sendo possível utilizar sobre escrita e mensagens urgentes;
- Tasklets e temporizadores, para priorizar a execução de eventos assíncronos (tasklet) ou tarefas não bloqueáveis dirigidas por tempo (temporizadores). Temporizadores permitem uma implementação tendenciosa das políticas flexíveis de tempo sem utilizar uma tarefa completamente da RTAI;
- Tratamento de interrupções em espaço de usuário (*User Space Interrupt* - USI), permitindo a implementação de gerenciadores de interrupções e drivers diretamente em espaço de usuário, facilitando seu desenvolvimento e teste.

Estes serviços são implementados através da *Application Programming Interface* (API) da RTAI, que somente poderá ser utilizada após a inicialização de módulos RTAI.

4.3.1.1 Arquitetura RTAI

Para a RTAI, o sistema operacional Linux funciona como uma tarefa de prioridade mínima, realizando operações somente quando não houver nenhuma operação de tempo real, provendo a capacidade de realizar tarefas em tempo real, independente de outras tarefas que o Linux esteja realizando (Figura 19).

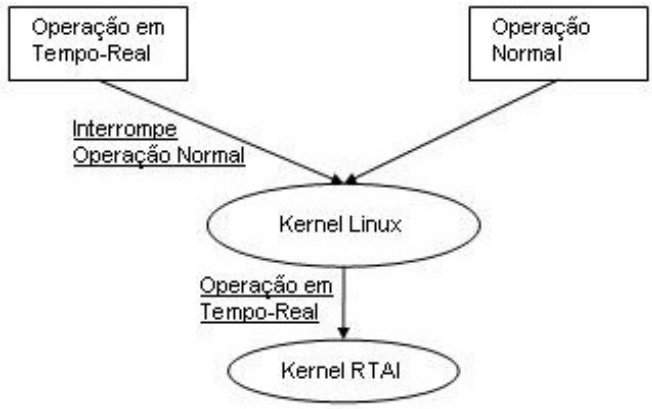

Figura 19 Interação kernel Linux/kernel RTAI

Assim, o kernel de tempo real efetivamente intercepta interrupções de hardware antes que estas cheguem ao kernel do Linux. Logo, o kernel do Linux não possui controle direto sobre a habilitação e desabilitação de interrupções. Quando o Linux tenta desabilitar interrupções, o kernel de tempo real apaga a bandeira de habilitação (*enable flag*) de um software interno de interrupção, mas as interrupções continuam habilitadas, ou seja, a Tabela de Descrição das Interrupções (*Interruption Description Table* - IDT) será apagada, porém as interrupções ainda estarão habilitadas (ABBOT, 2003).

Quando ocorrer uma interrupção de hardware, o kernel da RTAI primeiramente determinará quem atenderá esta interrupção. Se for uma tarefa de tempo real, esta será agendada. Se for uma tarefa do Linux, a IDT será checada. Se estiver habilitada, o manipulador específico de interrupções será invocado. Se estiver desabilitada, será anotado que a interrupção ocorreu e entregue posteriormente quando forem re-habilitadas as interrupções.

Para realizar a comunicação entre processos de tempo real e processos Linux a RTAI disponibiliza a implementação de FIFOs e compartilhamento de memória específica.

4.3.1.2 API e módulos RTAI

O uso de módulos permite a expansão do Linux quando houver necessidade. Como um *patch* do Linux, a RTAI também é baseada em módulos. Assim, para conseguir funcionamento em tempo real, faz-se necessário a inicialização de alguns módulos da RTAI. Esses módulos devem ser inicializados de acordo com a necessidade do usuário, entretanto

alguns são fundamentais e devem ser iniciados em ordem especifica, pois outros módulos dependem destes.

Através destes módulos, o kernel da RTAI poderá atender as responsabilidades de sistemas operacionais de tempo real, descritas na seção 3.2.1. Os módulos da RTAI têm como função (CLOUTIER et al., 2000):

- Inicia todas as estruturas e variáveis de controle do RTAI;
- Cria uma cópia da IDT e das entradas de endereços dos manipuladores de *Interrupt Request* (IRQ) do Linux;
- Inicia as funções especificas que gerenciam as interrupções.

A API da RTAI oferece várias funções para construção, gerenciamento e destruição de tarefas de tempo real, assim como o tratamento de interrupções externas. Esta API é distribuída entre os 22 módulos da RTAI. Durante o processo de configuração, cada característica pode ser habilitada ou desabilitada separadamente. A seguir algumas funcionalidades principais são apresentadas, assim como os módulos que fornecem estas funcionalidades (FRANKE, 2007):

- Tratamento de interrupções de tempo real e serviços de clock (rtai_hal);
- Gerenciamento das interrupções em nível de usuário (rtai_usi);
- Serviços de escalonamento:
    - Políticas de escalonamento preemptivo, prioridade externas fixas, round robin e FIFO na mesma classe de prioridade;
    - Escalonamento de tempo real rígido das tarefas de tempo real leves da RTAI (rtai_sched);
    - Escalonamento de tempo real rígido de todos os objetos agendáveis do Linux (rtai_lxrt);
- Transferência de dados inter processos:
    - Caixa de mensagens (rtai_mbx);
    - Mensagens (rtai_msg);
    - Filas de mensagem RTAI (rtai_tbx);
    - Filas de mensagem como POSIX (rtai_mq);
    - Memória compartilhada (rtai_shm);
    - FIFOs de tempo real (rtai_fifos);
    - Net RPC para comunicação em rede em tempo real (rtai_netrpc);

- Sincronização inter processo:
    - Semáforos (rtai_sem), suporte para heranças de prioridade máxima;
    - *Event flags* (rtai_bits);
    - Sinais (rtai_signal);
    - Tasklets, utilizadas quando funções precisam ser utilizadas em nível de usuário e kernel (rtai_tasklet);
- Drivers de tempo real para porta serial (rtai_serial);
- Suporte para ponto flutuante no kernel (rtai_math);
- Suporte para utilização de LEDs (rtai_leds);

Mais informações sobre o funcionamento desta API e dos módulos RTAI podem ser encontradas em (RACCIU; MANTEGAZZA, 2006).

4.3.2 Matlab/Simulink/RTW e RTAI (RTAI-LAB)

O RTAI-LAB é uma aplicação que explora a própria camada *middleware* da RTAI. Esta ferramenta provê um sistema para o desenvolvimento, construção, execução e monitoramento de quaisquer controladores simples/multi tarefa e simuladores de tempo real baseados na RTAI, quer especificamente codificados em uma linguagem de alto nível, tipicamente C++, ou gerado automaticamente, tanto pelo proprietário Matlab/Simulink/RTW, quanto pelo o gratuito Scilab/Scicos. O código em tempo real do alvo pode ser gerado e executado local/distribuidamente, assim como a interface de monitoração pode ser executada local/remotamente. A interface para a aplicação de tempo real permite a alteração de parâmetros ajustáveis, visualização e anotação das atividades dos sinais selecionados e supervisão de desempenho (DOZIO; MANTEGAZZA, 2003).

O RTW é um gerador de código automático da linguagem C para Simulink. Utilizando o Simulink é possível criar, simular, e analisar sistemas dinâmicos híbridos complexos (contínuo/discreto, linear/não linear) através da conexão de blocos funcionais, disponíveis por várias bibliotecas pré-configuradas, com uma interface gráfica amigável. Uma das grandes vantagens do RTW consiste em seu gerador de código completamente configurável que especifica como transformar um modelo de blocos do Simulink em código C, permitindo a

geração de um programa que possa ser executado, independente de sistema operacional ou plataforma, permitindo assim não somente geração de código, mas ainda realização de simulações de tempo real com hardware externos conectados ao computador, como servomotores, por exemplo, (DOZIO; MANTEGAZZA, 2003).

O Matlab/Simulink, em sua distribuição para o Linux, foi selecionado para o desenvolvimento de experiências devido sua estabilidade e confiabilidade. Além disto, por ser amplamente utilizado no meio acadêmico, principalmente por estudantes das áreas de sistemas de controle, sendo, inclusive, caracterizado como uma ferramenta computacional padrão para aplicações destas áreas. Utilizando o Matlab/Simulink podem-se desenvolver experiências virtuais e reais, utilizando vários *toolboxes* para aplicações de controle, disponíveis com o software (CASINI et al., 2004). Na figura 20 pode-se observar a integração entre Matlab/Simulink e RTAI, gerando o RTAI-LAB.

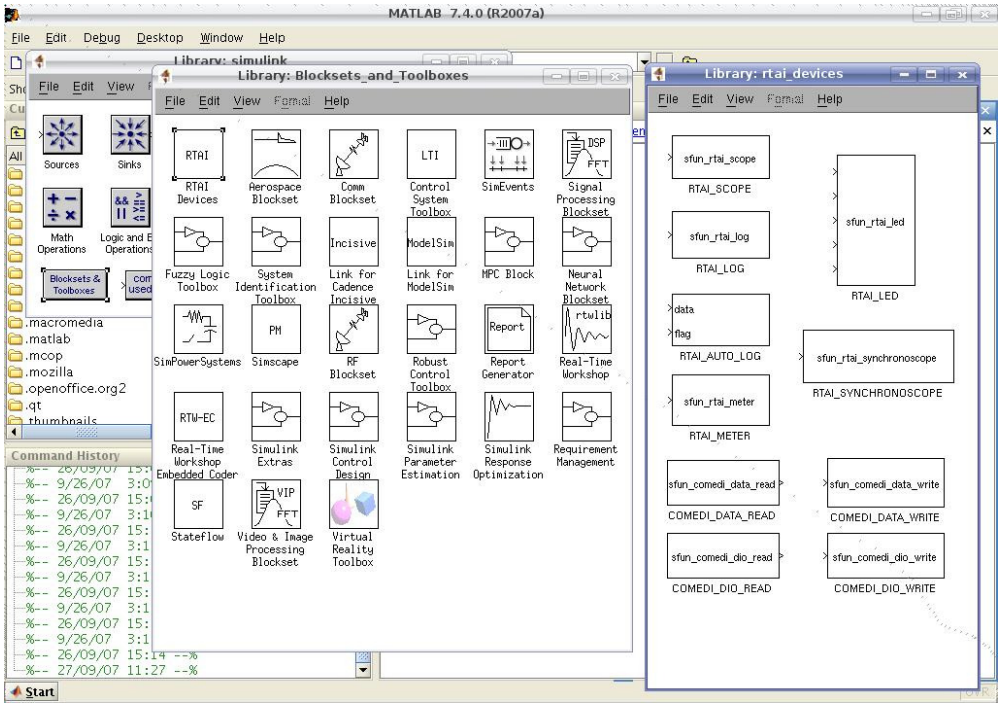

Figura 20 Integração Matlab/Simulink e RTAI

Após o desenvolvimento da experiência, o RTAI-LAB disponibiliza uma interface para acessá-la tanto local quanto remotamente, a xrtailab (figura 21). Para acesso remoto através do RTAI-LAB, os módulos que formatam, transmitem e recebem mensagens e pacotes, na camada de transporte, são completamente integrados dentro do código de tempo real através de uma camada *middleware* de tempo real, chamada net_rpc, sem a necessidade de adaptação e conexão com outra implementação da camada de transporte. Esta camada

integra aplicações locais ou distribuídas simplesmente adicionando um nó e uma porta identificadora em qualquer função da RTAI, ou seja, o nó e a porta da máquina remota que executará a função. O suporte disponível atualmente é através do UDP. Apesar de ser um protocolo Ethernet não determinístico, garante performances aceitáveis para utilização de um hardware de alta velocidade, para larguras de banda baixas ou médias, em controladores distribuídos (DOZIO; MANTEGAZZA, 2003).

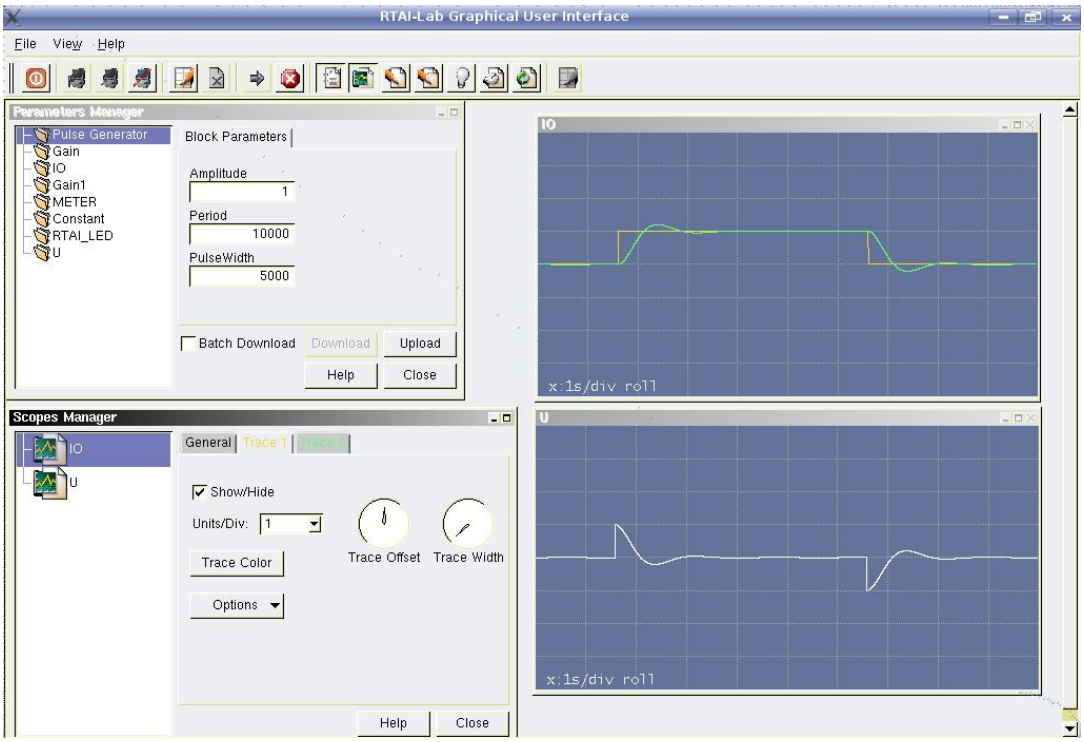

Figura 21 xrtailab

O acesso remoto ao LabExp poderia ser realizado através da arquitetura apresentada até o momento, entretanto todos os clientes que acessarem o laboratório precisam possuir, pelo menos a xrtailab instalada. Do ponto de vista da facilitação do acesso, assim como encapsulamento dos procedimentos referentes à instalação das tecnologias, não se pretende requerer ao usuário a instalação de nenhum software, ou manipulação do sistema operacional para utilização deste laboratório, por isso, foram utilizada as ferramentas do projeto RTAI-XML.

4.3.3 RTAI-XML

O projeto RTAI-XML é um componente servidor do projeto RTAI, que implementa um serviço orientado para o desenvolvimento de aplicações de controle em tempo real. Este projeto iniciou para suprir a necessidade de um grupo do *Dipartimento di Sistemi e Informatica* da Università degli Studi di Firenze – Itália, principalmente focado em possuir uma plataforma flexível para o aprendizado de desenvolvimento de sistemas de controle, permitindo aos estudantes testarem seus programas remotamente, através da internet. A primeira versão do RTAI-XML apresentou potencial impacto para a idéia básica da separação de tarefas de tempo real rígido e leve na lógica de programação. A separação entre tempo real rígido e leve, para o desenvolvimento de uma interface de tempo real para interação entre homem-máquina, pode ser facilmente visualizada, referindo-se a tempo real rígido como a implementação de comunicação com hardware e algoritmos de controle, e tempo real leve ao desenvolvimento da interface do usuário e análise e manipulação de dados (BASSO et al. 2005).

Assim, o servidor RTAI-XML é instalado em uma máquina onde uma experiência de tempo real está em execução e permanece aguardando solicitações de um cliente para interação com a experiência. O programa do cliente interage com a experiência através da rede TCP/IP, utilizando um protocolo baseado em XML, possibilitando a monitoração do processo de tempo real e alteração dos parâmetros de tempo real. Em outras palavras, o servidor RTAI-XML proporciona ao usuário um mecanismo para acessar experiências de controle remotamente, adicionando flexibilidade a RTAI. Basicamente esta arquitetura é composta pelo módulo do Servidor, pela interface Servidor-Experiência e pela comunicação entre Servidor-Cliente (BASSO et al., 2005).

4.3.3.1. Arquitetura RTAI-XML

O objetivo da RTAI-XML é obter uma estrutura onde um processo de tempo real (alvo) e os procedimentos de interação do usuário (cliente) são executados isoladamente, configurando a comunicação de dois CPUs diferentes através da internet. Desta forma, dois

computadores independentes permanecem interligados através de uma rede de computadores, compartilhando recursos de hardware, software e dados, caracterizando um sistema distribuído (TANENBAUM; STEEN, 2007). A figura 22 representa a arquitetura da RTAI-XML.

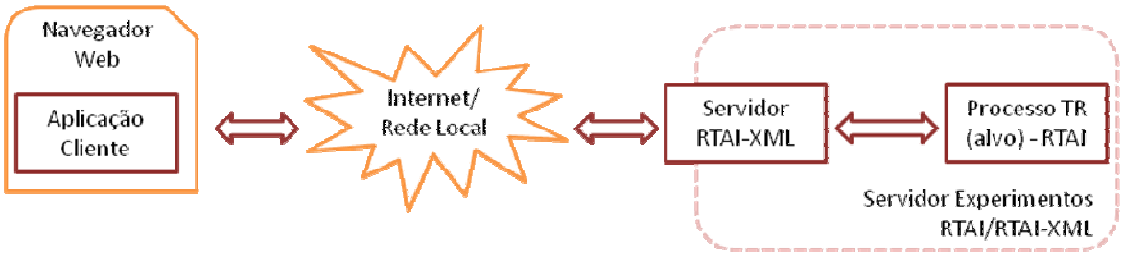

Figura 22 Arquitetura RTAI-XML
Fonte: (RTAI-XML, 2009)

O ponto principal do sistema é o servidor RTAI-XML, que funciona como uma ponte entre os domínios de tempo real rígido (alvo) e leve (cliente), permitindo instanciar o procedimento de tempo real leve da rede da internet para a rede de tempo real rígido RTnet, disponível na RTAI. Pelo lado da internet, o RTAI-XML implementa um servidor *Remote Procedures Call* (RPC) baseado em XML, dentro do domínio RTAI, conectando-se diretamente ao alvo utilizando chamadas de tempo real. Basicamente, esta arquitetura é composta por três módulos (BASSO et al., 2005):

a) processos em tempo real (alvo): estão situados dentro do domínio RTAI e funcionam em tempo real rígido. Eles podem ser gerados utilizando técnicas apresentadas anteriormente no RTAI-LAB ou utilizando a API da RTAI;
b) servidor RTAI-XML: localizado no domínio RTAI, entre a aplicação alvo e o cliente. O servidor gerencia as aplicações disponíveis, conectando e desconectando-as do domínio, obtendo a estrutura e alterando os parâmetros em tempo real. Geralmente, ele supervisiona a comunicação com processos de tempo real. Por outro lado, disponibiliza uma interface externa que permite a interação com diferentes alvos dentro domínio RTAI. Esta interface é baseada em XML através de HTTP e é estruturada como um serviço web que provê informações sobre o domínio RTAI-XML, o estado de cada alvo registrado e uma série de RPC que são utilizadas para monitorar e controlar as aplicações alvo em execução;
c) aplicações externas: são clientes dos serviços do servidor RTAI-XML. Essas aplicações não são desenvolvidas no ambiente de tempo real e podem ser

implementadas em qualquer linguagem de programação, comunicando-se com o servidor RTAI-XML utilizando XML-RPC.

Uma opção para aplicação externa, que é a utilizada pelo LabExp, é o aplicativo *jrtailab*, desenvolvido pela equipe do projeto RTAI-XML. Este aplicativo foi desenvolvido utilizando a linguagem de programação Java implementado para um cliente XML-RPC genérico. A figura 23 apresenta o aplicativo em execução no LabExp. O cliente pode acessar e utilizar esse applet sem haver necessidade de download do aplicativo, o único requisito do sistema é um navegador web e a realização da instalação do JRE (*Java Run-time Environment*), disponível gratuitamente na página da Sun Microsystems (SUN, 2009). A principal vantagem deste aplicativo é controlar um alvo registrado no domínio RTAI-XML, permitindo o estabelecimento de conexão com o processo de tempo real, observação do sinal escolhido, verificação e atualização dos parâmetros do processo e supervisão do processo em execução.

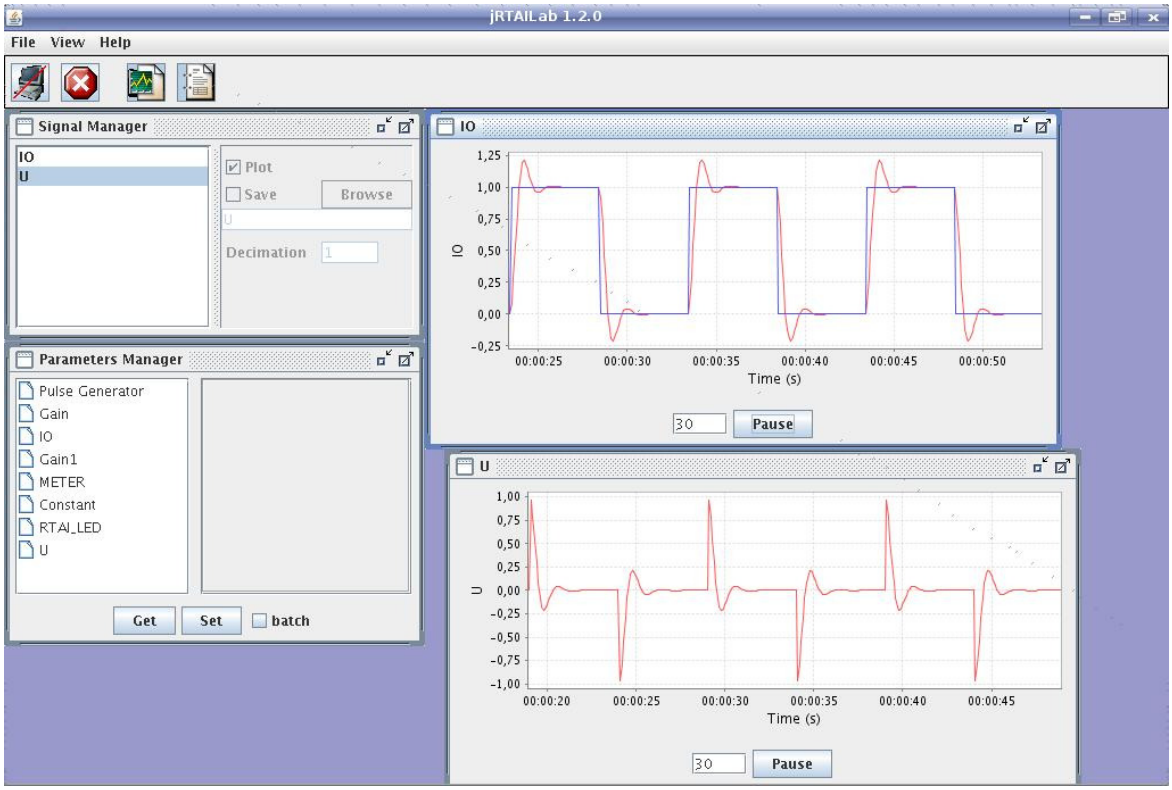

Figura 23 jrtailab em execução

A interface disponibilizada pelo servidor RTAI-XML é baseada em XML-RPC (XML-RPC, 2009) que utiliza XML para representar dados e HTTP/TCP como protocolo de

comunicação. As etapas de comunicação entre o cliente e o servidor, utilizando XML-RPC, são estas (LAURENT et al., 2001):

a) o programa cliente realiza uma RPC utilizando o cliente XML-RPC, especificando o nome do método, parâmetros e um servidor-alvo;
b) o cliente XML-RPC recebe o nome do método e os parâmetros e os empacota como um XML. Em seguida o cliente emite uma requisição HTTP **post,** contendo as informações requeridas pelo servidor-alvo;
c) o servidor Web*,* no servidor-alvo, recebe a requisição **post** e envia o conteúdo XML a um escutador XML-RPC;
d) o escutador XML-RPC analisa o conteúdo XML visando obter o nome do método e os parâmetros. Em seguida, chama o método apropriado e envia seus parâmetros;
e) o método retorna uma resposta ao processo XML-RPC, permitindo ao processo empacotar a resposta como XML;
f) o servidor Web retorna aquele XML como resposta a requisição **post**;
g) o cliente XML-RPC analisa o conteúdo XML para extrair a resposta e envia para o programa do cliente;
h) o programa do cliente continua o processamento com a resposta obtida.

A sobrecarga gerada, devido a XML e HTTP, é enviada através da rede aumentando a necessidade de largura de banda, entretanto sem aumentar a informação entregue. Por isso, serviços web não são adequados para comunicação que a quantidade de dados que uma aplicação de tempo real necessita. Para solucionar este problema, o servidor RTAI-XML separa a comunicação em dois níveis (BASSO et al., 2005). Assim, quando o servidor RTAI-XML necessita atender a requisição de um cliente, ele disponibiliza duas alternativas: se a requisição envolver informações gerais sobre o estado do domínio ou somente a obtenção do sinal e/ou parâmetros das estruturas, para uma determinada aplicação, então esta resposta utilizará a mesma tecnologia de requisição, ou seja, XML/HTTP/TCP. Alternativamente, se a requisição envolver elevada taxa de dados, então o servidor será alternado para outro nível de comunicação. Este outro nível foi desenvolvido explorando um protocolo especifico diretamente através do TCP, fazendo com que a sobrecarga gerada seja reduzida e a transmissão, para largura de banda disponível, otimizada.

4.4 DESENVOLVENDO UM EXPERIMENTO:

Abaixo segue um roteiro para desenvolvimento de aplicações em tempo real, utilizando o RTAI-LAB. Ressalta-se que todos os servidores devem estar devidamente configurados e inicializados:

a) Desenvolvimento do diagrama de blocos utilizando Simulink. Este diagrama representará o sistema implementando e a estratégia de controle escolhida. O diagrama será composto por blocos da própria biblioteca, juntamente com blocos específicos da RTAI e para Aquisição Digital, quando houver necessidade, provendo uma interface para monitoração do RTAI-LAB e suporte para Entrada/Saída, respectivamente. A RTAI e as bibliotecas Comedi provêem blocos específicos para aquisição digital de dados;

b) Geração de Código em C. Nesta etapa o usuário deverá somente selecionar o modelo para geração de código correto para que a linguagem de compilação do RTAI-LAB possa gerar automaticamente o programa de controle. Em seguida, o código gerado deverá ser compilado através da utilização do arquivo criado e a interface específica do RTW deve ser conectada ao sistema do RTAI-LAB, chamado rt_main, para tornar o alvo de tempo real executável;

c) O arquivo final pode ser executado com uma série de opções que permitem escolher entre os modos de tempo real leve/rígido, execução periódica/one shot, temporizador interno/externo, tempo final de execução finito/infinito, além de outras opções. Este arquivo final pode ser acessado através do xrtailab.

d) Para acesso remoto, mantendo as características de tempo real, o usuário poderá utilizar o jRtaiLab, ou qualquer aplicação cliente desenvolvida para comunicar-se com o servidor RTAI-XML, utilizando XML-RPC. Utilizando o jRtaiLab, duas etapas devem ser realizadas:

   – Desenvolvimento de um script para comunicação com o servidor RTAI-XML. Este script deverá conter alguns parâmetros que são fundamentais para o funcionamento da aplicação. Um exemplo deste script é apresentado abaixo:

      - target=ballandbeam #nome do experimento

- dir=/usr/local/rtai-xml/experimentos/ #caminho para o diretório onde o experimento está localizado;
- priority=8 #prioridade de tempo real;
- verbose=true #gerar relatório;
- rts=RTS #scope de tempo real
- rte=RTE #Led digital de tempo real;

– Informar, no jRtaiLab, o endereço do servidor do experiências, a porta de comunicação e o nome do experimento, criado no script.

Seguindo estes procedimentos, experimentos de tempo real, simulados ou reais, poderão ser criados e disponibilizados para acesso remoto através do laboratório, utilizando as tecnologias RTAI e RTAI-XML.

5 MODELAGEM DO LABORATÓRIO

Nos capítulos anteriores foram definidas a concepção do LabExp e as tecnologias que utilizadas no LabExp. Neste capítulo será apresentado o processo de modelagem, sendo este considerado uma aplicação web, e o desenvolvimento das interfaces do LabExp.

A distinção entre sites web e aplicações web está no uso da lógica do negócio. A lógica do negócio é composta por regras e processos que formam o estado do negócio do sistema, ela é composta de validações de campos e sistemas de dados (CONALLEN, 2003). Aplicações web são, de fato, sistemas de informática projetados para utilização através de um navegador web, tanto na internet quanto na intranet, sendo estes sistemas executados em um servidor web. A tabela 2 apresenta algumas diferenças entre sites e aplicações web:

Tabela 2

Sites Web x Aplicações Web

Sites Web	Aplicações Web
Geralmente está disponível somente através internet.	Geralmente está disponível através da intranet de uma determinada empresa e para um grupo de usuários cadastrados, entretanto, em alguns casos, estes usuários possuem acesso através da internet.
Não pode ser implementado como uma aplicação desktop.	Possui as mesmas funcionalidades de uma aplicação desktop.
Pode ser acessado por qualquer pessoa.	Somente pode ser acessado por usuários cadastrados.
Possui somente páginas estáticas.	Contem páginas estáticas e dinâmicas, sendo que as dinâmicas são construídas através de dados obtidos em um banco de dados central.
Para seu desenvolvimento é necessário, basicamente, conhecimentos de web design.	Trata-se de um sistema complexo e por isso são necessários conhecimentos avançados dos paradigmas de programação.

Devido à complexidade que envolve o desenvolvimento de uma aplicação web, acredita-se que é de fundamental importância realizar a modelagem desta aplicação. Estes

modelos simplificam a realidade em diversos níveis de abstração, facilitando o entendimento de procedimentos complexos. Além da compreensão, a modelagem funciona como um mecanismo de comunicação, permitindo que um grupo se comunique com outro em uma linguagem comum (CONALLEN, 2003).

As interfaces estáticas e dinâmicas do LabExp foram modeladas e desenvolvidas de acordo com princípios, padrões e recursos da *Unified Modeling Language* (UML), aplicados a aplicações web. A UML foi proposta por Grady Booch, Ivar Jacobson e James Rumbaugh como uma notação padrão para análise e desenvolvimento de aplicações orientado a objetos (BOOCH et al., 1997). Utilizando seus recursos é possível descrever todos os aspectos importantes referentes a um determinado sistema, entretanto devem-se haver critérios quanto a sua utilização visando evitar complexidade desnecessária, modelando somente aspectos importantes da aplicação.

Ressalta-se que a UML não possui recursos suficientes para modelagem de aspectos específicos de aplicações web, sendo assim utilizou-se a Web Application Extension for UML (WAE). Considerações sobre esta extensão são apresentadas na seção 5.1. A modelagem de uma aplicação web é importante, mais ainda tratando-se de uma aplicação utilizada para educação. A modelagem implica diretamente no processo de desenvolvimento da aplicação e um desenvolvimento ruim pode acarretar em desestímulo, desorientação e desinteresse dos usuários.

O processo de modelagem consistiu na análise dos requisitos necessários para realizar o desenvolvimento do laboratório, que serão apresentados na seção 5.2. Em seguida, na seção 5.3, será apresentado o processo de modelagem propriamente dito. Finalmente as interfaces desenvolvidas serão apresentadas na seção 5.4.

5.1 UML/WAE

Conforme especificado anteriormente, o LabExp é considerado uma aplicação web, sendo assim suas interfaces estáticas e dinâmicas foram desenvolvidas de acordo com padrões, princípios e recursos da UML aplicada a aplicações web.

A UML em seu estado original, não possui estereótipos suficientes para modelar aspectos específicos de aplicações web. A Web Application Extension for UML (WAE),

extensão para a UML criada por (CONALLEN, 1999), é utilizada para modelar o LabExp. A WAE estende a notação UML trazendo novos estereótipos com semântica e restrições adicionais, que permitem a modelagem de elementos específicos da arquitetura envolvida numa aplicação web incluindo-os nos modelos do sistema. Utilizando esta extensão é possível representar páginas, links e o conteúdo dinâmico no lado cliente e no servidor. A tabela 3 apresenta alguns ícones estereotipados desta extensão.

Tabela 3

Ícones estereotipados WAE.

Página servidora	
Página Cliente	
Página JSP	

Antes de iniciar o processo de modelagem, foi necessário realizar a análise dos requisitos do LabExp.

5.2 ANÁLISE DE REQUISITOS

Segundo (CONALLEN, 2003), os requisitos podem ser divididos em funcionais (RF) e não funcionais (RNF). Para nosso problema, os requisitos a seguir serão considerados e qualificados:

a) Camada do usuário:

- O sistema deverá permitir que usuários acessem o LabExp utilizando somente um navegador web, sem a necessidade de instalar ou utilizar qualquer outro aplicativo, a não ser o JRE (Java Runtime Environment), necessário para execução de applets Java (RF);

- Será permitido ao usuário interagir com experimentos através de applet Java específico. Neste applet o usuário poderá alterar parâmetros do experimento e observar a resposta em tempo real (RF);

- O sistema permitirá que usuários enviem arquivos de experimentos desenvolvidos utilizando o software Matlab/Simulink, segundo roteiro especifico, para que estes possam ser adicionados ao LabExp (RF);
- Questionário:
  - Após a execução dos experimentos e a navegação pelo LabExp, o usuário deverá responder um formulário sobre a utilização deste LabExp (RNF);
  - Todos os campos do formulário deverão ser preenchidos (RF);
- Os usuários poderão acessar um fórum para que possam discutir os resultados de seus experimentos, assim como outros assuntos relevantes (RNF);

b) Camada do servidor:

- O sistema deverá realizar controle de acesso dos usuários mediante login e senha específica (RF);
- Experimentos:
  - O sistema deverá disponibilizar aos usuários applet Java para que possam interagir com experimentos (RF);
  - O sistema deverá informar se o usuário está conectado ou não ao servidor de experiências (RF);
  - O sistema deverá informar se o servidor de experiências não estiver em funcionamento (RF);
- O sistema somente deverá permitir que os usuários enviem arquivos, verificando se os arquivos são válidos (desenvolvidos com o software Matlab/Simulink ou documentos de texto, com tamanho máximo de 03 megabytes). Se forem inválidos, deverá notificar o usuário (RF);
- O sistema deverá disponibilizar aos usuários um questionário, para que estes possam emitir opiniões e prover feedback aos administradores/professores (RF);
- Este questionário deverá ser implementado utilizando a linguagem JSP e os campos deverão ser validados utilizando JavaScript (RNF);

- O sistema deverá disponibilizar um fórum para que os usuários troquem informações entre si e com administradores/professores (RF).

Requisitos específicos referente ao desenvolvimento das interfaces do LabExp foram considerados, entretanto, como estes requisitos não podem ser verificáveis não foram determinados anteriormente. Estes requisitos são referentes principalmente a questões de usabilidade e design. Quanto à usabilidade, o sistema deveria possuir navegação simples e intuitiva, possuir somente conteúdo relevante e apropriado para web, clareza na arquitetura, entre outros. Em relação ao design, princípios estéticos, artísticos e éticos deveriam ser considerados (DZENDZIK, 2004).

5.3 PROCESSO DE MODELAGEM

A estratégia básica para a modelagem do LabExp foi desenvolver dois modelos centrais que serviriam como referência para o desenvolvimento de modelos mais concretos. Esta estratégia foi adotada por permitir o desenvolvimento de modelos com diferentes níveis de complexidade de acordo com a finalidade e o nível de conhecimento do desenvolvedor. Estes dois modelos são o diagrama de casos de uso e o diagrama de classes.

Um diagrama de casos de uso mostra um conjunto de casos de uso e atores e seus relacionamentos. Estes diagramas são utilizados para ilustrar as necessidades um sistema, observável por um ator. Através dos elementos dos diagramas de casos de uso é possível uma visualização de quem são os atores do sistema e quais os tipos de interação que o sistema permite (BOOCH et al., 1998).

Na figura 24 pode-se observar o diagrama de caso de uso superior para a LabExp. Nesta figura são observados os atores (Aluno e Administrador) e os casos de uso (Acessar Ambiente, Disponibilizar Experimento, Acessar Experimento, Enviar Arquivos, Emitir Opinião e Acessar Fórum). O administrador pode ser o professor ou tutor de um determinado grupo de alunos.

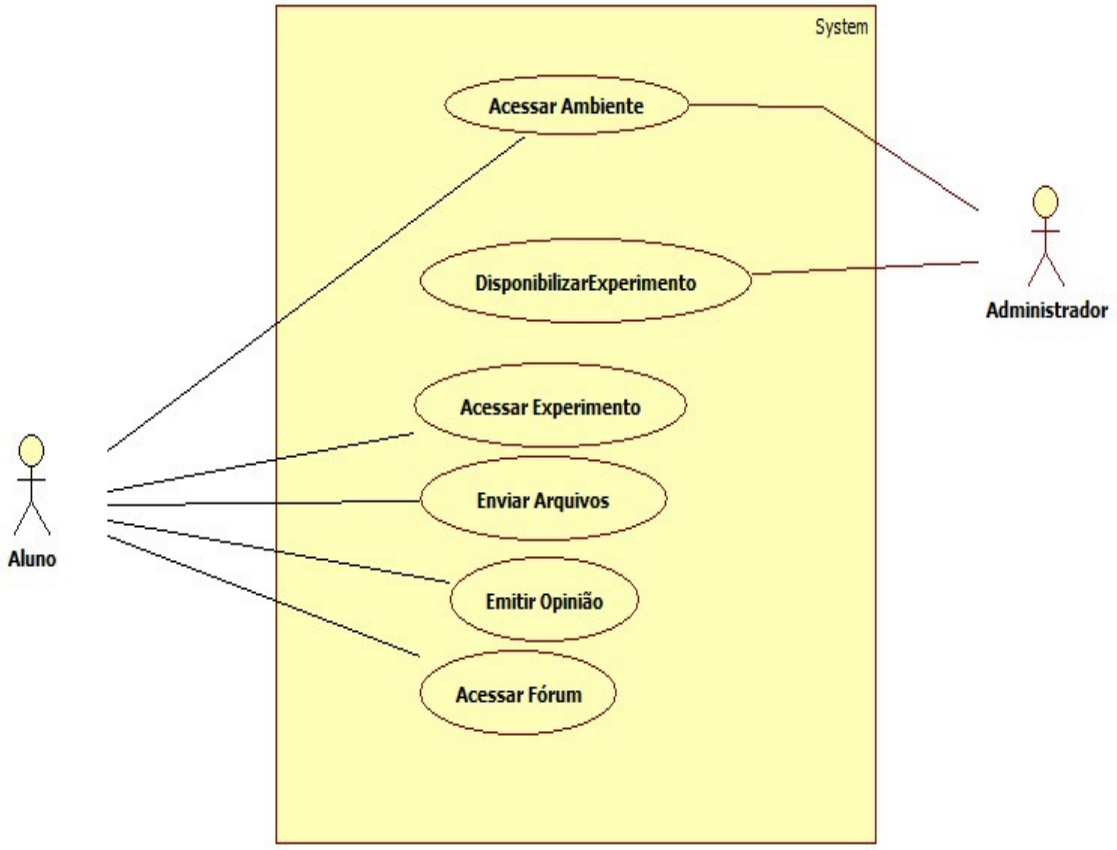

Figura 24 Caso de Uso superior

O diagrama de classes mostra um conjunto de classes, interfaces, colaborações e seus relacionamentos. Através de um diagrama de classes representa-se a estrutura de um sistema; a modelagem da visão estática e os serviços que o sistema deverá fornecer aos usuários finais (BOOCH et al., 1998). A figura 25 apresenta o diagrama de classes geral do sistema. Nesta figura podem-se observar os três servidores (web, experiências e aplicações) apresentados no capítulo 4.

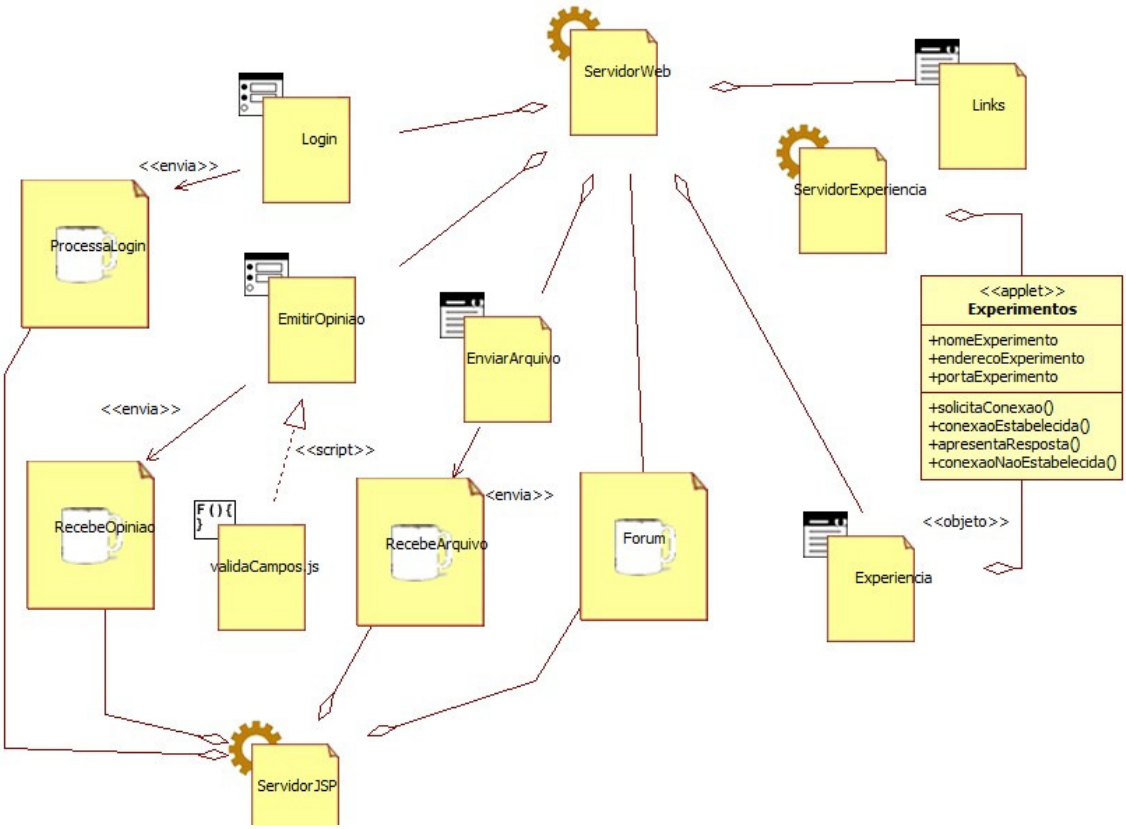

Figura 25 Diagrama de Classes Geral

Baseados nos diagramas de casos de uso e de classes pôde-se iniciar a modelagem mais concreta do LabExp. A UML provê diagramas de seqüência, colaboração, atividades, componentes, entre outros, com o objetivo de contemplar todos os aspectos de desenvolvimento referentes a uma determinada aplicação. Para o desenvolvimento do LabExp, a modelagem mais concreta foi realizada através dos diagramas de seqüência e atividades, acreditando-se que utilizando estes diagramas foi possível abordar os aspectos mais relevantes, sem adicionar demasiada complexidade ao processo de desenvolvimento.

Os diagramas de seqüência são utilizados na UML para a modelagem dos aspectos dinâmicos do sistema, como a interação de objetos que desempenham uma operação ou parte da funcionalidade do sistema. Estes diagramas são utilizados para mostrar períodos durante os quais os objetos desempenham as suas ações. Diagramas de atividades também modelam aspectos dinâmicos do sistema. Este processo envolve a modelagem das etapas do processo computacional. Com este diagrama pode-se modelar como um objeto se move de estado em estado em diferentes pontos de operação (BOOCH et al., 1998).

A seguir a modelagem mais concreta para cada caso de uso será apresentada.

5.3.1 Caso de Uso “Acessar Ambiente”

O primeiro caso de uso apresentado é o “Acessar Ambiente”. Este caso de uso apresenta um usuário acessando o ambiente do LabExp. Quando o usuário tenta acessar o ambiente o servidor web verifica o estado do usuário. Esta verificação consiste na utilização de *cookies* (fragmentos de dados que o servidor web solicita ao navegador web para manter e retornar cada vez que ele fizer uma solicitação subseqüente de um recurso HTTP). Se o usuário ainda possuir estado válido será apresentado ao mesmo à página inicial do ambiente. Se não, será solicitado que este usuário realize autenticação mediante inserção de login e senha. Esta medida foi adotada por questões de segurança e para realizar o controle acesso dos usuários, principalmente quando houver necessidade de limitar este acesso a um grupo específico de usuários. Esta verificação é realizada através do banco de dados para autenticação de usuários do servidor de aplicações (MOODIE, 2007). Se o login e senha forem válidos, será apresentado ao usuário a pagina inicial do ambiente. Se não, será apresentada uma página de erro e o usuário deverá realizar autenticação novamente.

Os diagramas de seqüência e atividades são apresentados nas figuras 26 e 27, respectivamente.

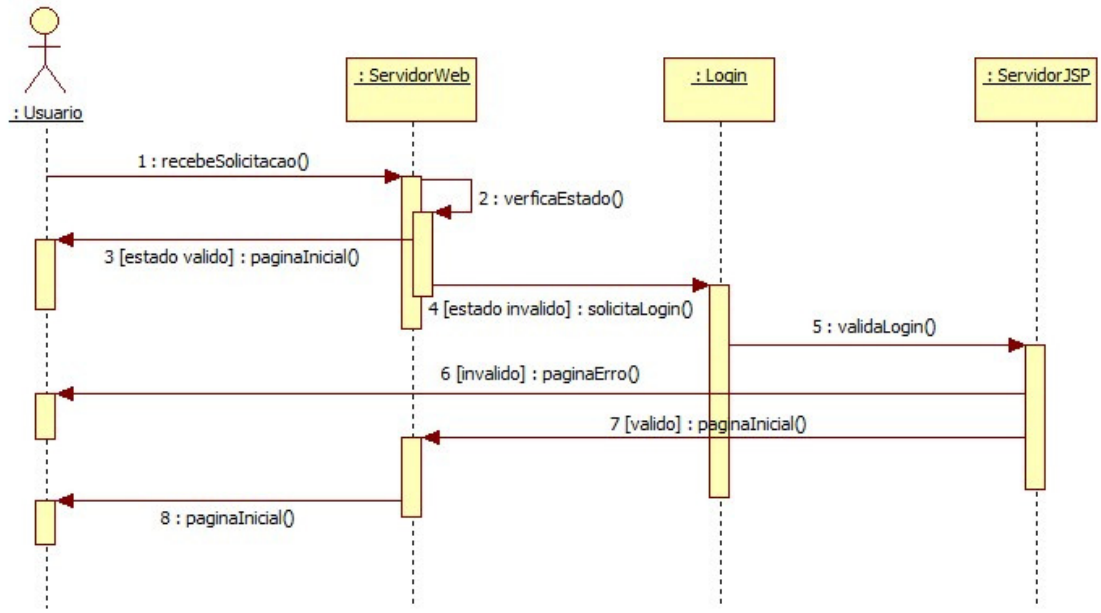

Figura 26 Diagrama de Seqüência para o caso de uso “Acessar Ambiente”

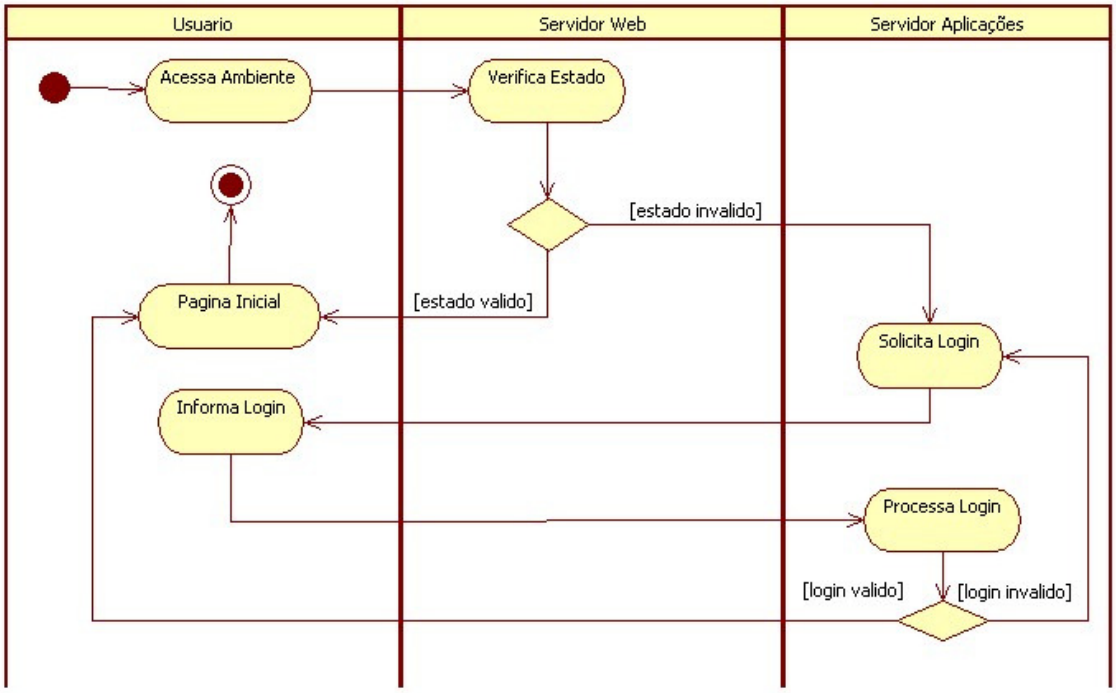

Figura 27 Diagrama de Atividades para o caso de uso "Acessar Ambiente"

Deve-se ressaltar que para os demais casos de uso considera-se que o usuário realizou autenticação no sistema. Se o usuário tentar acessar qualquer página sem realizar autenticação ou permanecer muito tempo ocioso, será solicitado que o mesmo realize login novamente. Ressalta-se também que se o usuário entrar com o endereço *Uniform Resource Locator* (URL) inválido no navegador será exibido, pelo servidor web, uma página de erro. Esta situação não foi modelada nos diagramas por se tratar de um comportamento padrão do servidor web.

5.3.2 Caso de Uso "Disponibilizar Experimento"

Este caso de uso apresenta o desenvolvimento de um experimento pelo administrador/tutor do LabExp. A maior parte deste processo acontece no servidor de experiências. Primeiramente o administrador deverá inicializar os módulos da RTAI e o servidor RTAI-XML. Após a inicialização dos módulos, o experimento será desenvolvido utilizando o software Matlab/Simulink e em seguida seu código de tempo real gerado. Nesta etapa o experimento estará disponível para acesso local através do xrtailab. Para disponibilizá-lo para acesso remoto, um script deverá ser desenvolvido. Para maiores informações sobre o

processo de desenvolvimento de experimentos utilizando o RTAI-LAB ver seção 4.4. Finalmente no servidor web será disponibilizado o applet jRtaiLab para cada experimento. As figuras 28 e 29, respectivamente, representam os diagramas de seqüência e atividades

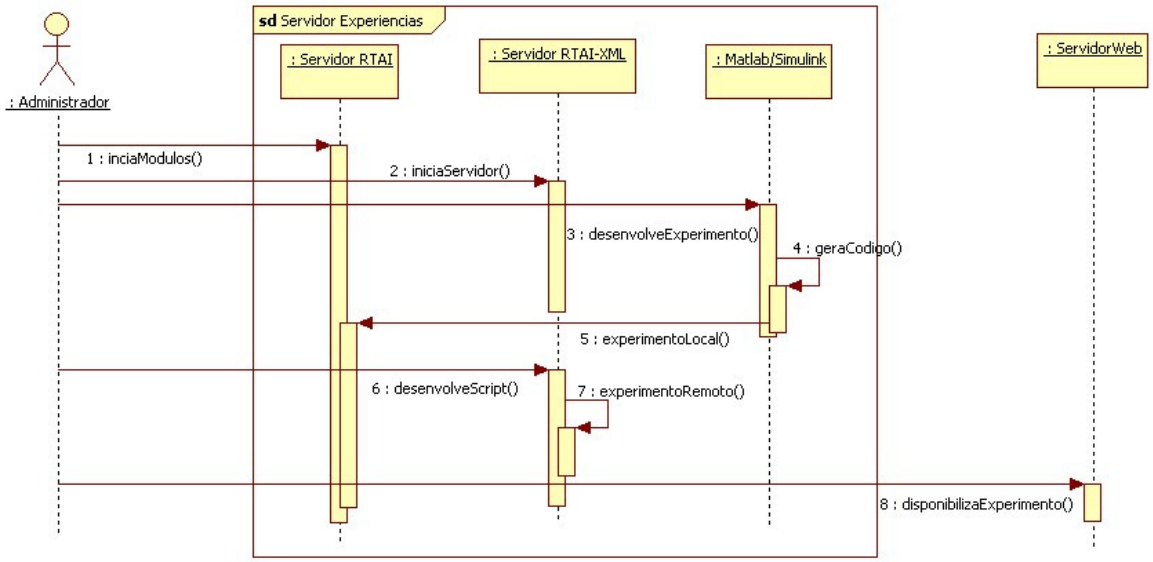

Figura 28 Diagrama de Seqüência do caso de uso "Disponibilizar Experimento"

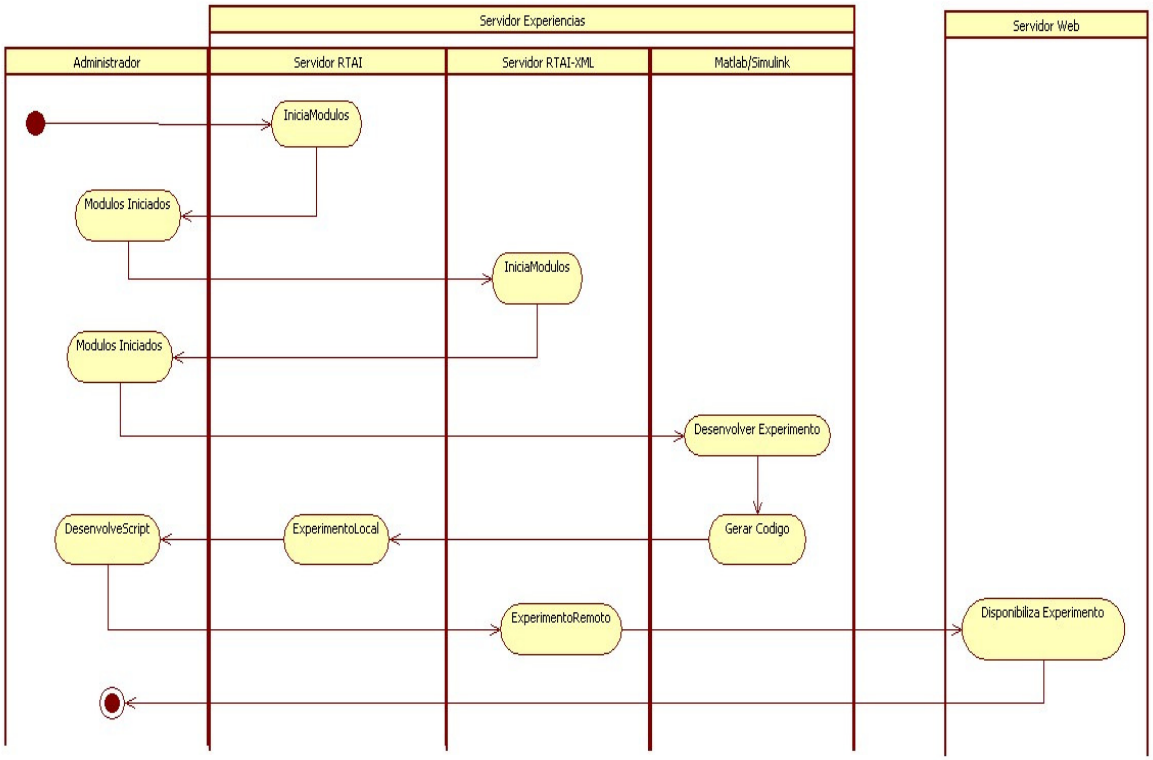

Figura 29 Diagrama de Atividades do caso de uso "Disponibilizar Experimento"

5.3.3 Caso de Uso “Acessar Experimento”

Neste caso de uso um usuário solicita ao servidor web, através de um navegador web padrão, uma página contendo os experimentos disponíveis para interação. O servidor, através do navegador, disponibiliza a página. Em seguida, o usuário poderá selecionar que experimento deseja interagir. Na página do experimento, é apresentado ao usuário um roteiro sobre aquele experimento, assim como o applet Java jRtaiLab, que permitirá a interação com este experimento. Através deste applet, será solicitada conexão com o servidor de experiências. Se houver algum erro, o jRtaiLab informará ao usuário e retornará para a página do experimento. Se não, a conexão será estabelecida e o usuário poderá alterar os parâmetros do experimento e observar a resposta.

Os diagramas de seqüência e atividade, para este caso de uso podem ser observados nas figuras 30 e 31, respectivamente.

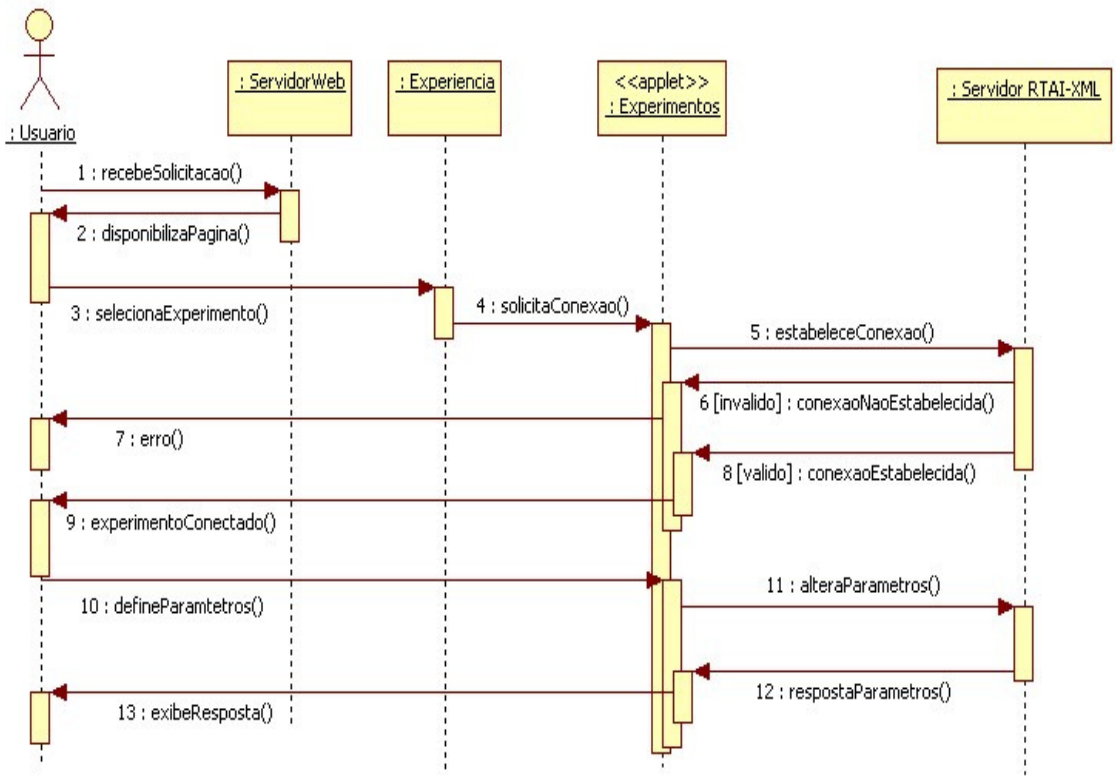

Figura 30 Diagrama de seqüência para o caso de uso “Acessar Experimento”

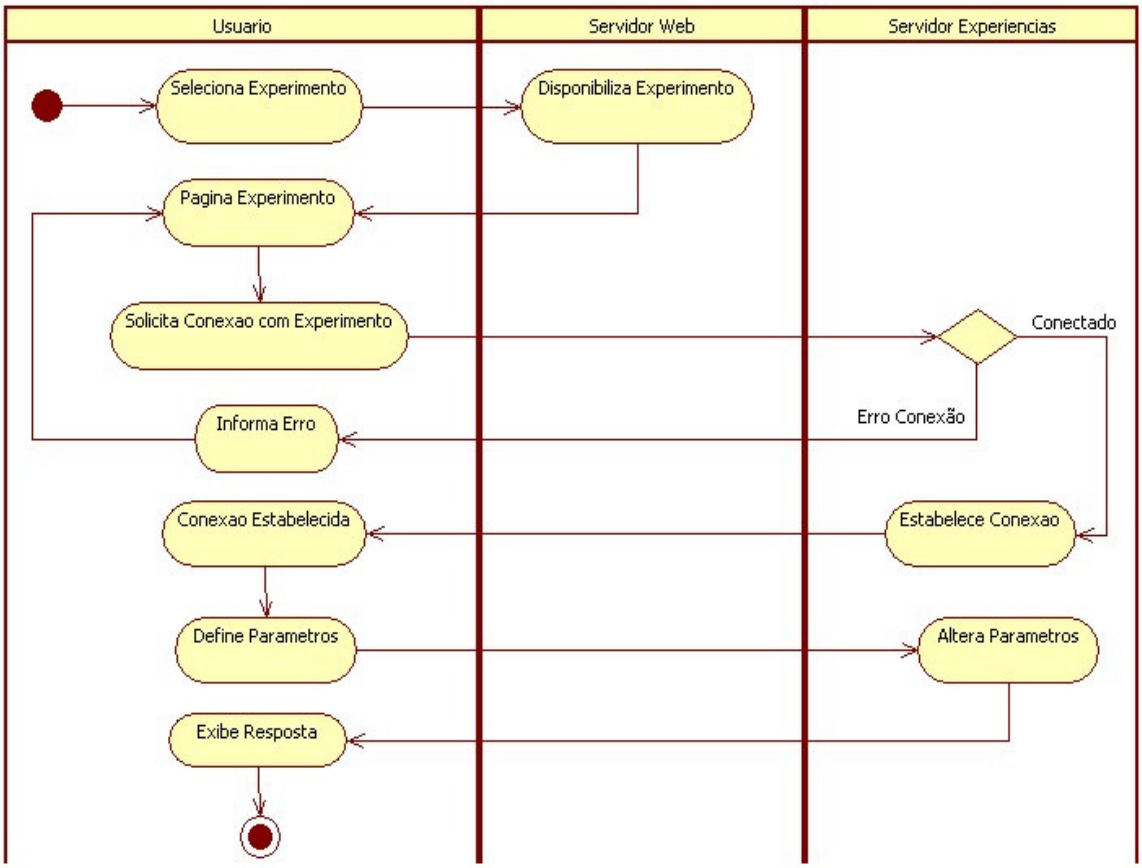

Figura 31 Diagrama de atividades para o caso de uso "Acessar Experimento"

5.3.4 Caso de Uso "Enviar Arquivos"

Neste caso de uso um usuário solicita ao servidor web, através de um navegador web padrão, uma página onde será possível enviar arquivos para o servidor. O servidor, através do navegador disponibiliza a página. Em seguida, o usuário poderá selecionar o arquivo que deseja enviar. A aplicação "recebeArquivo.jsp" verifica o arquivo. Este arquivo deverá possuir tamanho máximo de 03 megabytes e possuir a extensão "mdl" (desenvolvido no simulink) ou "doc" (documento de texto). A idéia é que o usuário envie seu experimento e um documento descrevendo este experimento. O servidor de aplicações processará a solicitação da aplicação. Se o arquivo for inválido, o usuário receberá uma notificação com a justificativa de erro e deverá selecionar outro arquivo. Se válido, o arquivo será enviado e o usuário notificado. As figuras 32 e 33 apresentam os diagramas de seqüência e atividades respectivamente.

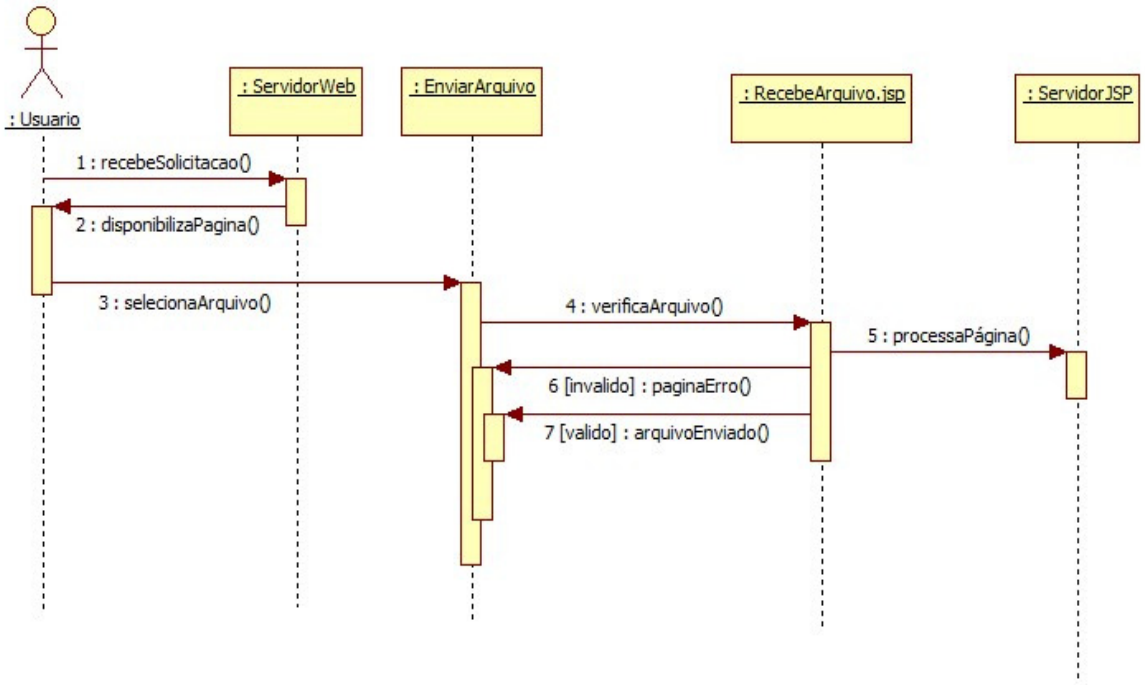

Figura 32 Diagrama de Seqüência do caso de uso "Enviar Arquivo"

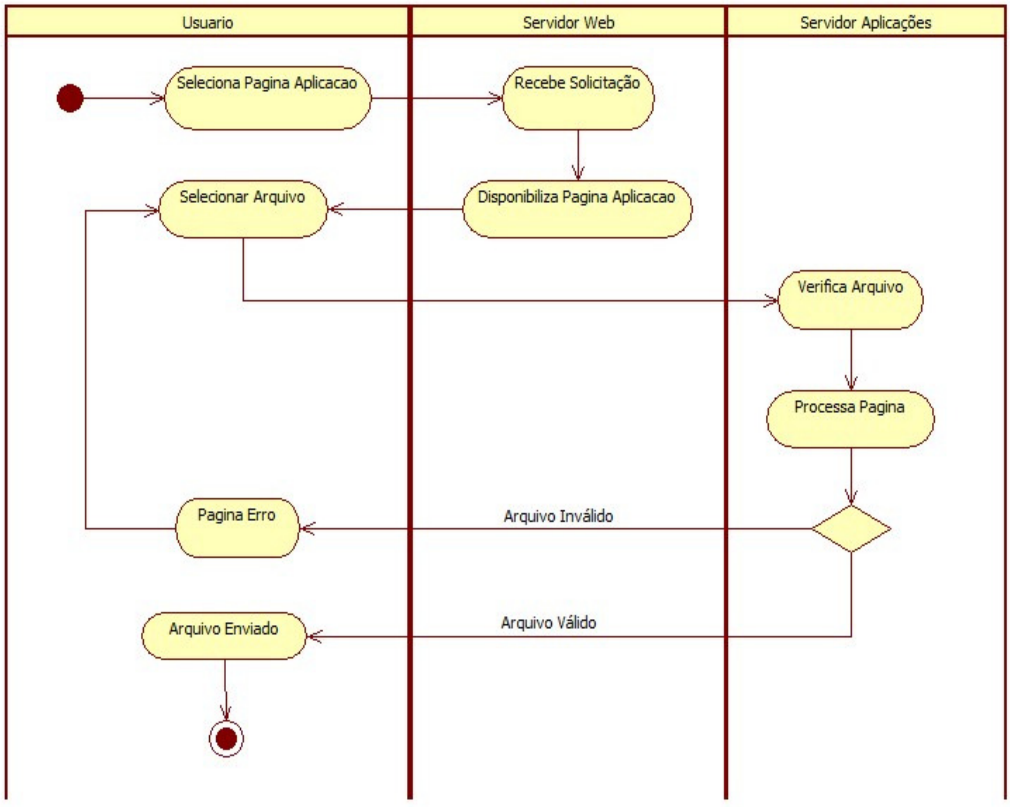

Figura 33 Diagrama de Atividades do caso de uso "Enviar Arquivo"

5.3.5 Caso de Uso “Emitir Opinião”

Neste caso de uso um usuário solicita ao servidor web, através de um navegador web padrão, uma página onde deverá emitir opiniões sobre o LabExp. O servidor, através do navegador disponibiliza a página. O usuário deverá responder ao questionário, preenchendo todos os campos corretamente. Após o preenchimento, o javascript “validaCampos.js” irá verificar se todos os campos foram preenchidos. Se não, o usuário receberá uma mensagem de erro e deverá preencher as informações pendentes. Se estiver correto, a aplicação “recebeOpinião.jsp” irá receber o formulário e o servidor de aplicações irá processá-lo. Assim o usuário irá receber uma mensagem que o formulário foi enviado com sucesso. A figura 34 apresenta o diagrama de seqüência e a figura 35 apresenta o diagrama de atividades.

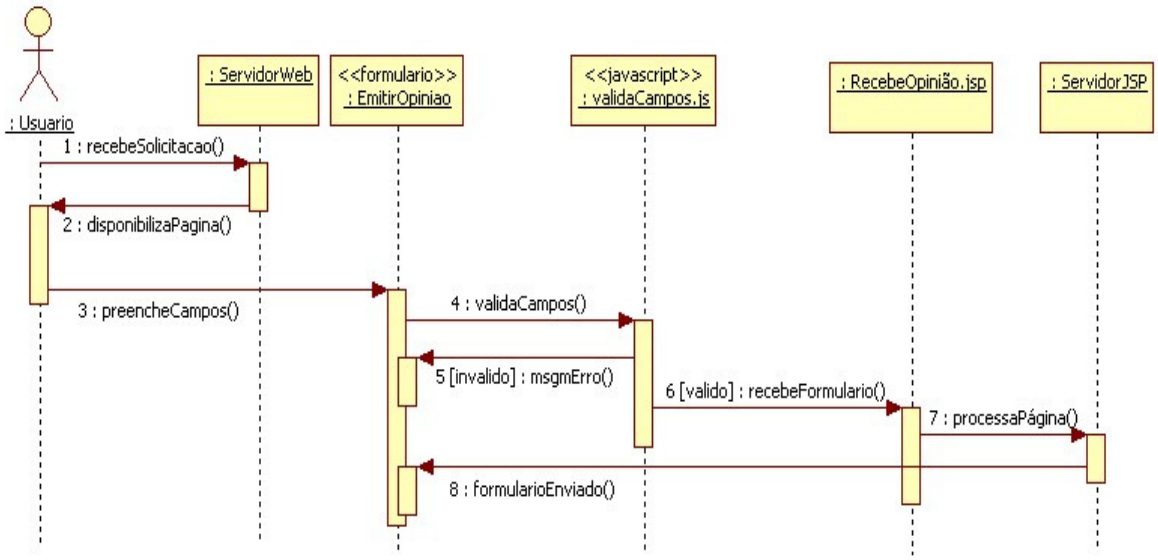

Figura 34 Diagrama de Seqüência para o caso de uso “Emitir Opinião”

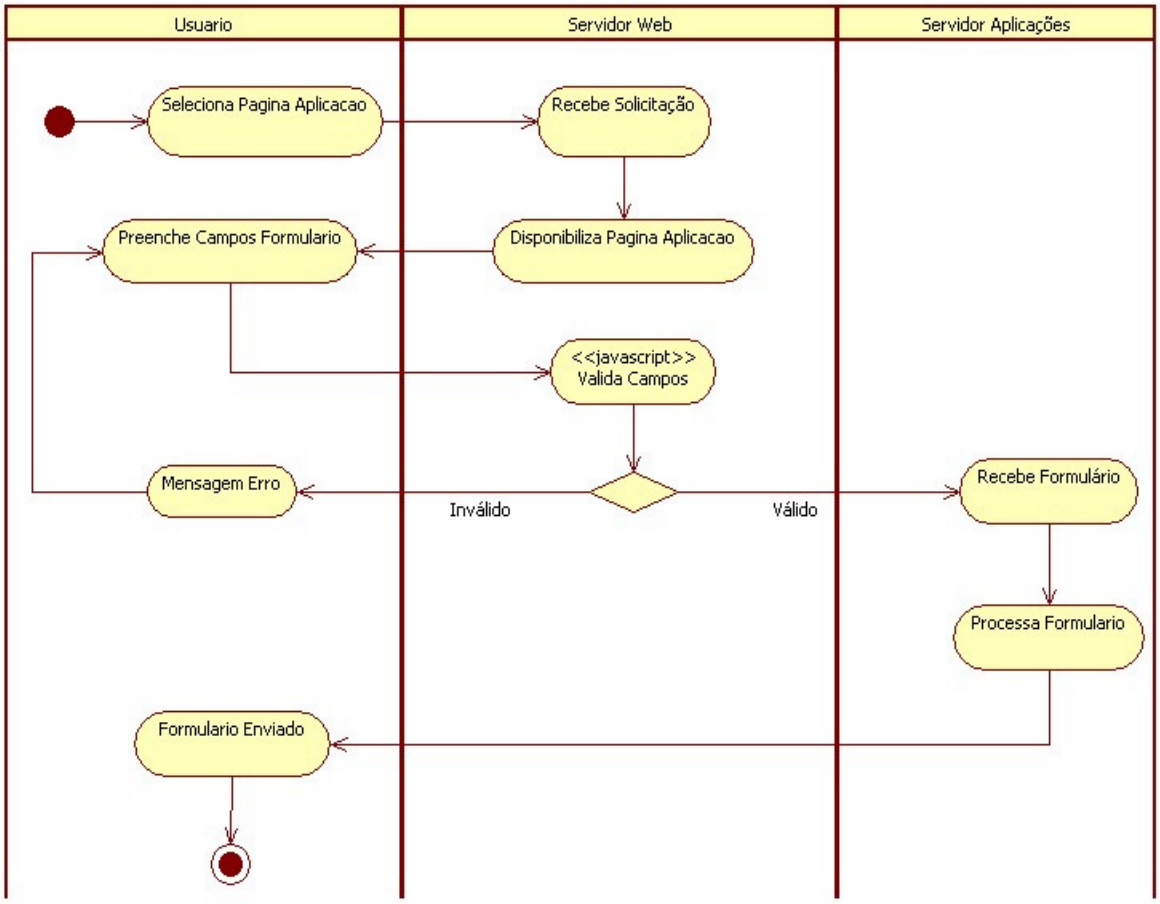

Figura 35 Diagrama de Atividades para o caso de uso "Emitir Opinião"

5.3.6 Caso de Uso "Acessar Fórum"

Neste caso de uso o servidor de aplicações irá processar as solicitações do usuário, pois o fórum é uma aplicação JSP. Sendo assim o usuário irá solicitar ao servidor de aplicações acesso ao fórum, através de um navegador padrão, e este servidor disponibilizará ao usuário acesso. Após acessar o fórum serão apresentados ao usuário os tópicos disponíveis para discussão e este usuário poderá postar mensagens. Os diagramas de seqüência e atividades são apresentados nas figuras 36 e 37, respectivamente.

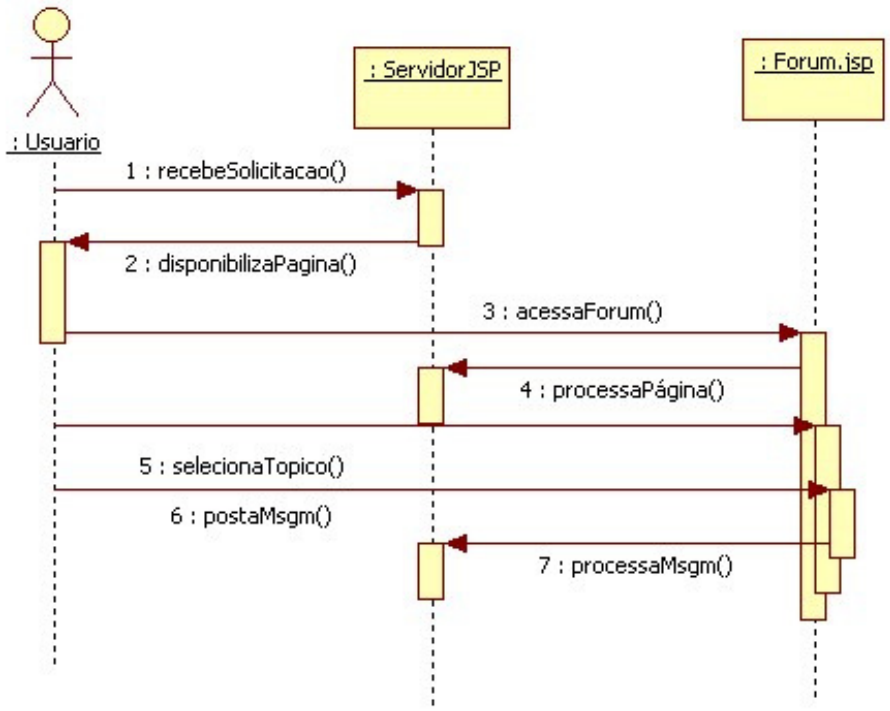

Figura 36 Diagrama de Seqüência para o caso de uso "Acessar Forum"

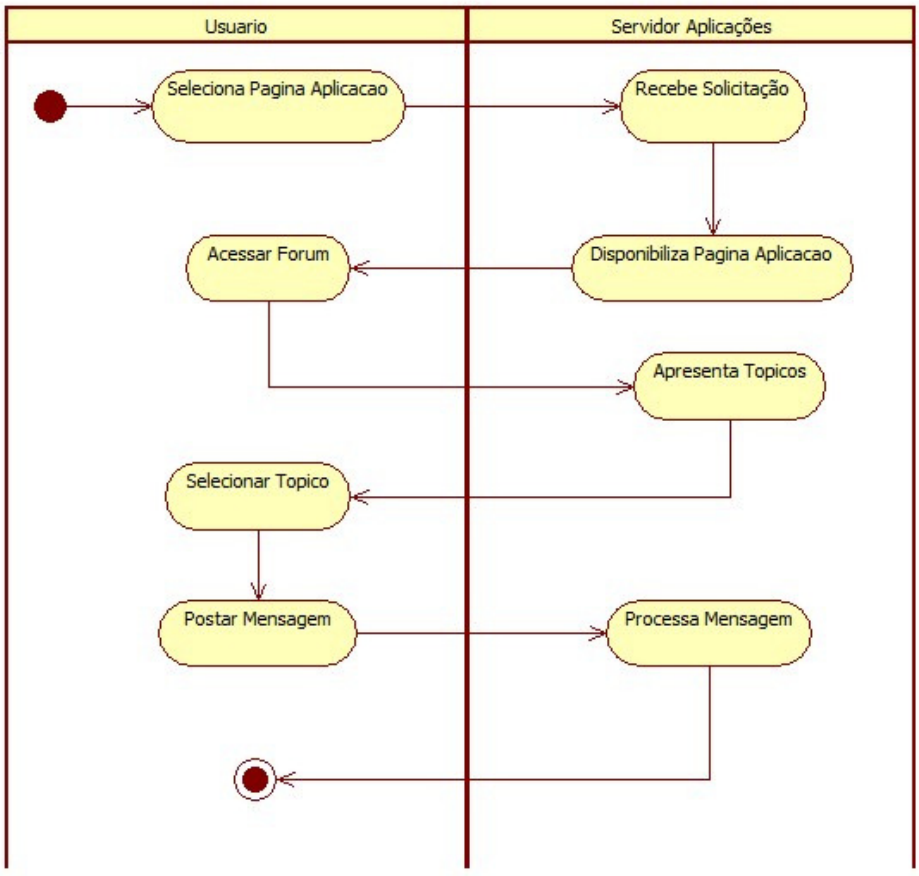

Figura 37 Diagrama de Atividades do caso de uso "Acessar Forum"

5.4 DESIGN DAS INTERFACES

A fase final no processo de desenvolvimento do LabExp foi o design das interfaces. A proposta para esta fase envolveu estética, arte e ética como base para um estilo de design. Esta fase foi considerada como artística considerando que após o levantamento dos requisitos, a definição das tecnologias e a modelagem do sistema já se obtiveram as funcionalidades desejadas (DZENDZIK, 2004).

Para o desenvolvimento das interfaces, os princípios básicos de design apresentados no capítulo 2 foram considerados:

- Características do usuário: os usuários do LabExp são alunos de graduação cursando disciplinas das áreas de sistemas de controle;
- Planejamento da interface;
- Navegação: foram adotadas técnicas que facilitam a navegação do usuário assim como o mantém sempre informado da sua localização;
- Elementos multimídia: são utilizados simulações e roteiros, descrevendo e orientando sobre o funcionamento de um determinado sistema.

Na figura 38 observa-se a interface inicial do ambiente. Nesta página é apresentada a finalidade fundamental do ambiente, assim como os links para acesso a outras áreas do laboratório. Vale ressaltar que para acessar o LabExp os usuários deverão possuir login e senha específica. Esta medida foi adotada por questões de segurança, pois a partir deste ponto o usuário irá interagir com os servidores do laboratório.

Figura 38 Página inicial do ambiente

Na figura 39 pode-se observar a página web onde as experiências disponíveis poderão ser acessadas. Nesta página, novos experimentos facilmente poderão ser adicionados e todos os experimentos disponíveis podem ser visualizados.

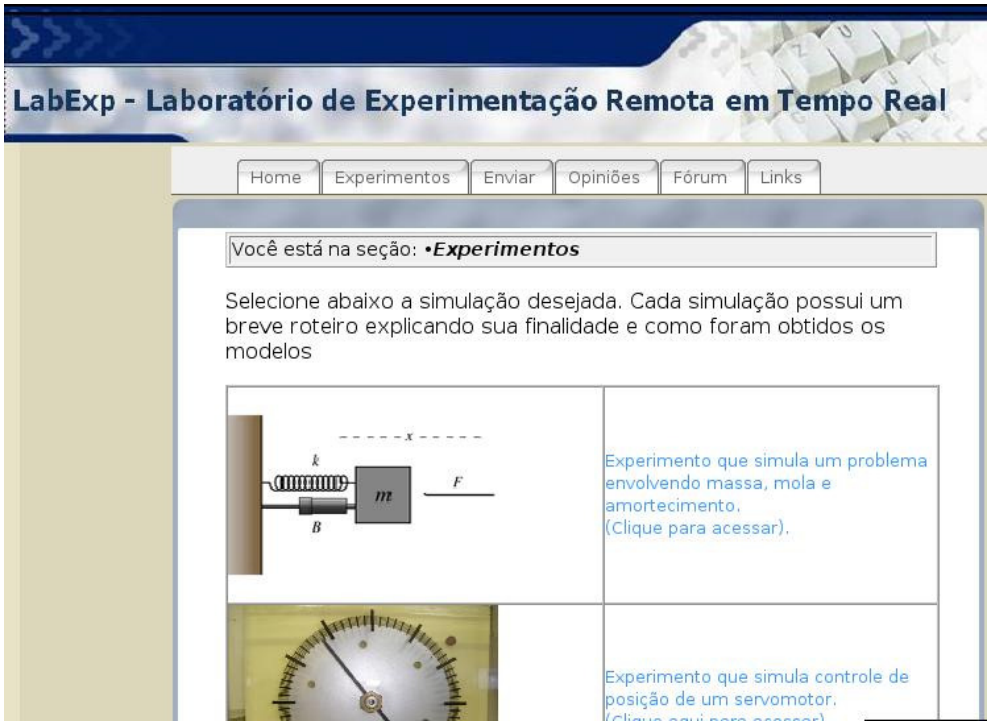

Figura 39 Experiências disponíveis

Atualmente existem três aplicações disponíveis no LabExp. Uma permite ao usuário enviar experimentos desenvolvidos utilizando o software Matlab/Simulink; a segunda envia as opiniões/questionários desenvolvidas por usuários; a terceira aplicação é um fórum implementado visando proporcionar aos usuários um mecanismo de interação, o JForum.

Conforme especificado anteriormente, estas aplicações foram desenvolvidas utilizando a linguagem de programação JavaServer Page (JSP) e esta linguagem precisa de um container para operar, sendo o Apache Tomcat o container utilizado.

A utilização destas aplicações permite que usuários compartilhem conhecimento e possam evoluir através de um aprendizado colaborativo, discutindo e comparando resultados com outros alunos, tutores e professores.

Um experimento enviado ao servidor será analisado e poderá ser disponibilizado no laboratório de experimentação remota para acesso/interação por outros usuários. Na aplicação para enviar experimentos (Figura 40) somente serão aceitos dois tipos de arquivos:

a) Arquivos desenvolvidos com a utilização do Matlab/Simulink, ou seja, com a extensão "mdl". Anteriormente, mencionou-se que o Matlab/Simulink é o CACDS escolhido para o desenvolvimento de experimentos, portanto o software utilizado no LabExp. Sendo assim, somente os modelos desenvolvidos com a utilização deste software serão aceito;

b) Documentos, ou seja, arquivos com a extensão "doc". Documentos serão aceitos devendo apresentar a descrição da experiência enviada, assim como qualquer informação relevante.

Outra restrição em relação ao envio de arquivos é referente ao tamanho dos mesmos, devendo possuir tamanho máximo de 03 megabytes. Quaisquer arquivos diferentes dos especificados ou com tamanho superior a 03 megabytes serão rejeitados pelo sistema.

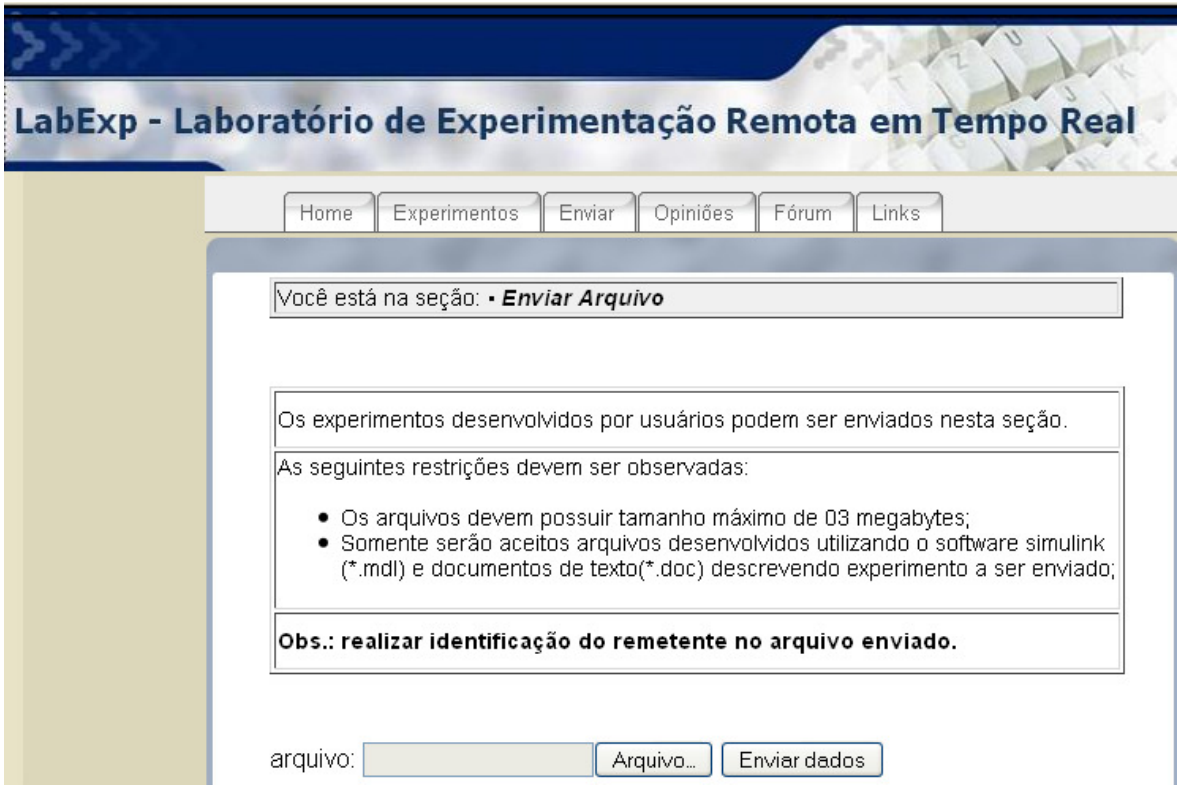

Figura 40 Enviar experimento

A figura 41 apresenta a interface que permite aos usuários emitir opiniões, participando com maior efetividade do laboratório, avaliando experiências, o ambiente geral e sugerindo novidades e melhorias. Estas opiniões também visam validar a utilização e mensurar o grau de interesse por usuários do LabExp. No momento em que o usuário enviar sua opinião, será criada uma pasta, no servidor, com o nome e sobrenome fornecido por este usuário e todas as informações fornecidas serão armazenadas em um arquivo de texto.

As opiniões e experimentos enviados poderão ser utilizados para melhorar/atualizar o ambiente, proporcionando aos usuários um ambiente de aprendizado colaborativo.

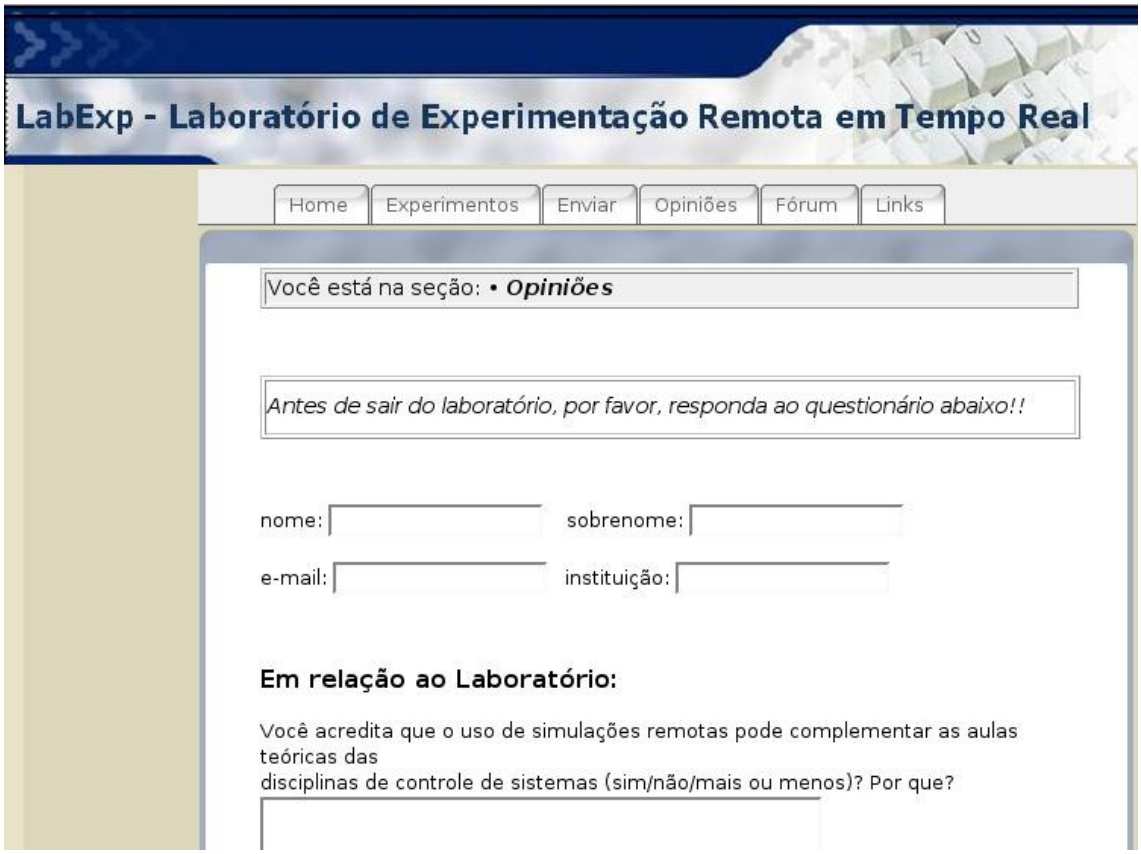

Figura 41 Opiniões

Semelhante ao envio de experimentos e opiniões, a utilização de um fórum proporcionará ao usuário o sentimento participativo no ambiente. Por isto foi implementado um fórum (Figura 42), o JForum.

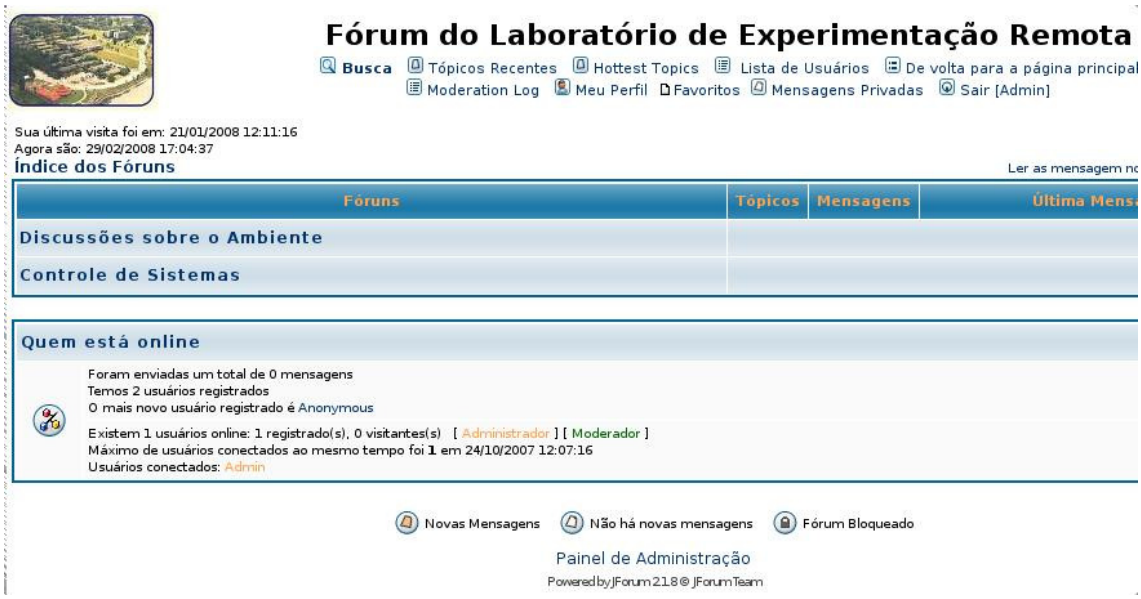

Figura 42 Fórum de discussão

Por fim, a figura 43 apresenta a seção “Links”, onde alguns links considerados interessantes, como o site da Universidade Federal do Pará, da RTAI e do RTAI-XML, são disponibilizados.

Figura 43 Links Úteis

6 RESULTADOS OBTIDOS

Neste capítulo serão apresentados os resultados obtidos através da utilização do LabExp por um grupo de alunos.

Atualmente existem três experimentos disponíveis no LabExp. Um experimento simula o controle de posição de uma massa, considerando que ela está conectada a uma mola e a existência de um amortecedor viscoso. O segundo experimento simula o controle de posição de um servomotor. O terceiro é um experimento bastante conhecido em disciplinas de sistemas de controle, o "Ball and Beam" (trad. Bola e Barra). A ação de controle neste sistema é regular automaticamente a posição da bola na barra através da alteração do ângulo desta barra. Este tarefa de regulação torna-se difícil considerando que a bola não permanece em somente uma posição na barra, mas movimenta-se com aceleração proporcional a inclinação da barra. Mais detalhes sobre os experimentos disponíveis podem ser encontrados no apêndice D.

Controlar um experimento, para este caso, refere-se a este sistema rastrear uma determinada referência de acordo com alguns parâmetros de desempenho, como tempo de subida, comportamento em regime permanente e resposta transitória.

Para controlar os dois primeiros experimentos foi utilizada uma estrutura de controle Proporcional, Integral e Derivativo (PID). Esta técnica consiste em aplicar ganhos proporcional, integral e derivativo, visando controlar o sistema. O ganho proporcional (Kp) deve reduzir o tempo de subida e reduzir o erro de regime permanente. O ganho integral (Ki) deve eliminar o erro de regime permanente, mas pode degradar a resposta transitória. O ganho derivativo (Kd) deve melhorar a estabilidade do sistema, reduzindo o overshoot e melhorando a resposta transitória do sistema (OGATA, 1985). Logo, os usuários podem alterar estes ganhos, assim como o valor de referência, e observar como os experimentos responderão a esta alteração.

No terceiro experimento adotou-se uma técnica de controle inteligente, o controle fuzzy. Um controlador fuzzy é uma técnica de controle baseada em informações heurísticas sobre o processo. Este controlador fornece uma metodologia formal para se representar, manipular e implementar o conhecimento humano heurístico sobre como controlar um sistema. A principal característica do controle fuzzy é que ele trabalha com regras lingüísticas, na forma *"Se...Então..."*, ao invés de modelos matemáticos e relacionamentos através de

funções (PARASKEVOPOULOS, 1996) e (WANG, 1997). Logo, usuários podem alterar as funções de pertencimento, assim como o valor de referência, e observar como o experimento responde a estas alterações.

Desta forma pode-se observar que o LabExp, através de seus servidores (web, aplicações e experiências), é capaz de executar experimentos utilizando distintas técnicas de controle, inclusive controle fuzzy. Os experimentos são acessado no LabExp através do jRtaiLab, conforme mencionado anteriormente.

Acessaram o laboratório estudantes de graduação do Programa de Educação Tutorial em Engenharia Elétrica (PET-EE) e estudantes de graduação, inclusive mestrado, do Laboratório de Recursos Hídricos da Universidade Federal do Pará, totalizando 20 estudantes. Depois de acessarem o LabExp, os usuários responderam um questionário, onde se pode mensurar seus graus de interesse, assim como suas opiniões e críticas sobre o laboratório.

O questionário foi divido em três partes. A primeira relacionada ao LabExp, a segunda referente ao applet jRtaiLab, utilizado nas simulações, e a última sobre o ambiente web do laboratório.

Relacionado ao laboratório, foram questionados se o laboratório poderia complementar o conteúdo das aulas teóricas de sistemas de controle e se as simulações implementadas contribuem para fixação do conhecimento adquirido. A figura 44 apresenta os resultados.

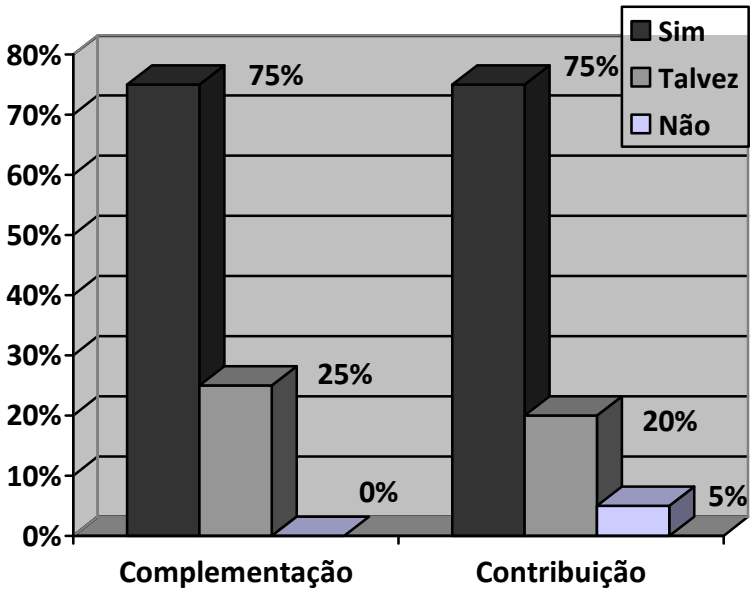

Figura 44 Questionário Laboratório

Sobre a utilização do applet jRtaiLab, os usuários foram questionados sobre a dificuldade em sua utilização, em alterar seus parâmetros e em interpretar suas respostas. A figura 45 apresenta os resultados.

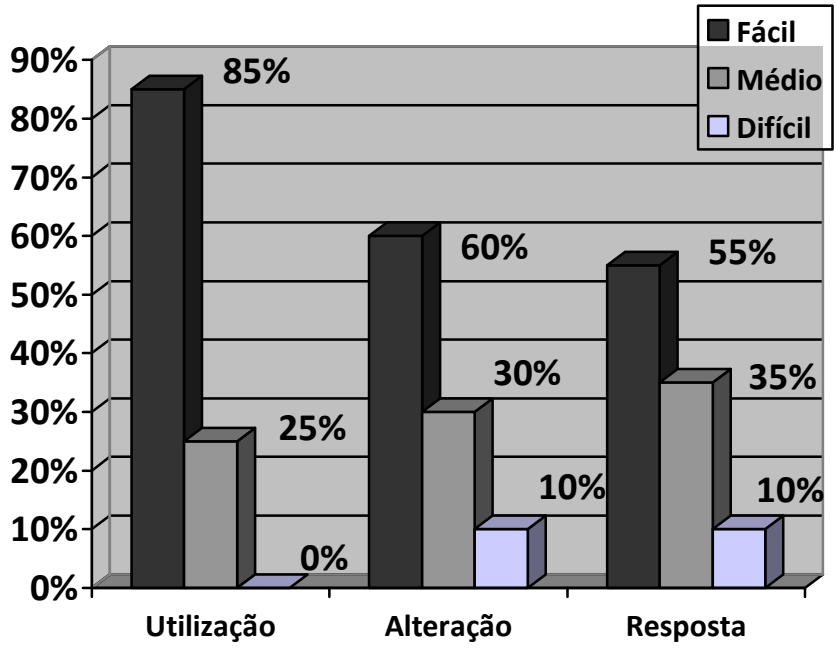

Figura 45 Questionário jRtaiLab

Relacionado ao ambiente do laboratório, responderam sobre o acesso e navegação no laboratório e sobre o acesso aos experimentos. Os resultados são apresentados na figura 46.

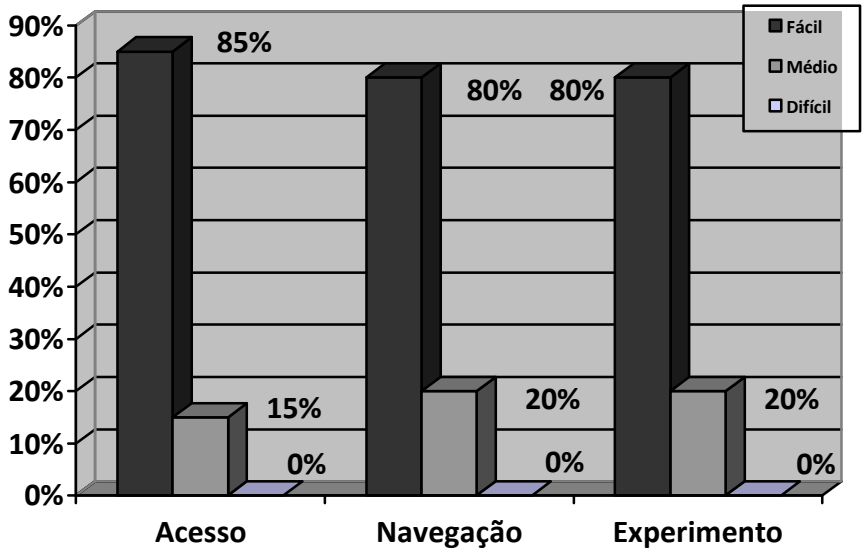

Figura 46 Ambiente LabExp

Finalmente os usuários responderam se preferiam trabalhar em um laboratório real, compartilhando recursos ou sozinhos utilizando o LabExp. Resultados na figura 47.

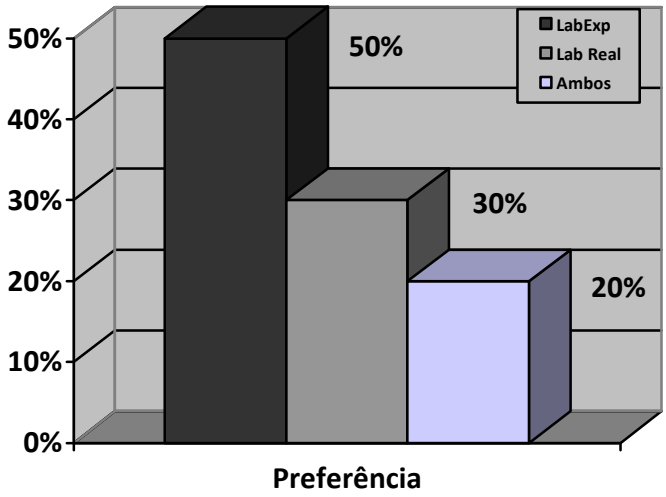

Figura 47 Preferência Usuário

7 CONCLUSÕES

Este trabalho propôs apresentar o processo de desenvolvimento de um laboratório de experimentação remota, sendo este utilizado para ensino de sistemas de controle. Este laboratório foi denominado LabExp e o seu processo de desenvolvimento consistiu na concepção do laboratório, definição das tecnologias utilizadas, processo de modelagem, sendo o mesmo considerado uma aplicação web, e o desenvolvimento das interfaces. Neste capítulo são apresentadas conclusões, dificuldades e sugestões para trabalhos futuros.

O objetivo geral do trabalho foi alcançado, ou seja, a implementação de um laboratório de experimentação remota para ensino de sistemas de controle, apresentando as tecnologias utilizadas e o processo de modelagem. Logo, referente à sua utilização em disciplinas de sistemas de controle, concluiu-se que:

- Sua implementação é viável, tanto do ponto de vista da educação online, quanto do ponto de vista das tecnologias utilizadas, pois houve aceitação majoritária dos alunos que utilizaram o LabExp, de acordo com os resultados obtidos, apresentados no capítulo 6;
- A utilização da RTAI proporcionou o comportamento desejado, pois os experimentos implementados funcionaram corretamente, tanto no servidor quanto na rede local;
- O LabExp pode funcionar como uma ferramenta auxiliar para professores e estudantes, tanto na rede local, quanto através da internet;
- Sua utilização proporcionaria uma forma de ensino mais dinâmica para a promoção de um aprendizado mais significativo, desenvolvendo a motivação pelo ensino-aprendizado, permitindo que o usuário desenvolva e envie seus próprios modelos, para que possam ser testados e utilizados por outros usuários do laboratório;
- A inclusão de novos modelos ou modificação dos modelos existentes seria imperceptível para o usuário, pois não se faz necessário a realização de download de nenhum software, sendo somente necessária a instalação do JRE.

Em relação ao processo de modelagem do LabExp, as seguinte conclusões foram obtidas:

- Constatou-se que para o desenvolvimento de um laboratório de experimentação remota é necessário o planejamento para modelagem de sistemas através de

abordagens, tais como a UML, que permitem maior consistência e clareza na definição da arquitetura. Sem esta modelagem o processo de desenvolvimento poderia demandar mais tempo e, consequentemente, implicar no desenvolvimento inadequado do laboratório;

- Com a utilização da WAE, pôde se realizar o tratamento das páginas clientes e servidoras, assim como das estáticas e dinâmicas, como classes, realizando sua diferenciação através de ícones (forma visual) e estereótipos (forma descritiva);
- O uso de ferramenta *Computer-Aided Software Engineering* (CASE) no processo de desenvolvimento foi essencial. Toda a modelagem foi realizada utilizando a ferramenta CASE StarUML (STARUML, 2008). Esta ferramenta possui código aberto, e pode modelar sistemas possuindo características semelhantes ao comercial IBM Rational Rose (IBM, 2008).

Em relação às dificuldades encontradas durante o processo de desenvolvimento do LabExp, duas podem ser destacadas:

- A primeira referiu-se a tentativa inicial de utilização do software Scilab/Scicos como ferramenta para desenvolvimento de aplicações do RTAI-LAB, almejando a utilização somente de softwares gratuitos. A dificuldade encontrada nesta etapa, demandando bastante tempo, referiu-se ao processo de configuração do Scilab para funcionar em conjunto com a RTAI, pois os parâmetros que poderiam ser modificados por usuários não apareciam corretamente. Durante esta tentativa, discussões com a equipe desenvolvedora deste software foram realizadas, entretanto sem sucesso, permanecendo inviável sua utilização até o momento;
- Outra dificuldade ocorreu durante o processo de configuração da RTAI. Conforme especificado no capítulo 4 a RTAI funciona como um patch para o sistema operacional Linux, sendo seus módulos inicializados de acordo com a finalidade do projeto. Este processo foi fundamental, pois, para aplicar este pacth no Linux fez-se necessário realizar a re-compilação do kernel do Linux, modificando suas configurações que, se realizadas incorretamente, podem implicar em mau funcionamento ou erro fatal no sistema operacional, impossibilitando sua recuperação.

Futuramente, pretende-se realizar algumas alterações para o laboratório, destacando:

- Realizar a utilização somente de ferramentas gratuitas no laboratório, ou seja, substituir o software comercial Matlab/Simulink pelo gratuito Scilab/Scicos;

- Desenvolver outros experimentos simulados, disponibilizando materiais teóricos com conteúdo mais extenso e explicativo, incorporando novas atividades realizadas em sala de aula. Pretende-se também desenvolver experimentos com sistemas reais;
- Implementar um Sistema Gerenciador de Conteúdo (*Content Management Systems* - CMS). Através de um CMS o administrador do LabExp poderia criar, editar e inserir conteúdos online, sem a necessidade de realizar programação de código, facilitando a criação, administração e publicação de todo o ambiente;
- Desenvolver outras aplicações multimídia com o objetivo proporcionar um ambiente mais dinâmico;
- Implementar o LabExp para uso regular em aulas da disciplina Laboratório de Sistemas de Controle no Curso de Graduação em Engenharia Elétrica.

REFERÊNCIAS

ABBOTT, D. **Linux for Embedded and Real-time Applications**. Burlington: Editora Elsevier Science, 2003.

ACT. **Automatic Control Telelab.** Disponível em <http://www.dii.unisi.it/~control/act/home.php>. Acesso em 30/04/2009.

AGUIRRE, L. A. **Introdução à Identificação de Sistemas**: Técnicas Lineares e Não-Lineares Aplicadas a Sistemas Reais. Belo Horizonte: Editora UFMG, 2004.

ALLY, M. Foundations of educational Theory for online learning. In: **Theory and Practice of Online Learning**. Canadá: Athabasca University, 2004, p. 3-31.

ALVES, J. R. M. **Educação a distancia e as novas tecnologias de informação e aprendizagem**. Artigo do Programa Novas Tecnologias na Educação, 1998. Disponível em <http://www.engenheiro2001.org.br/programas/980201a1.htm>. Acesso em 30/04/2009.

ANDERSON, T. Toward a Theory of online learning. In: **Theory and Practice of Online Learning**. Canadá: Athabasca University, 2004, p. 33-60.

APACHE. **HTTP Server Project.** Disponível em <http://httpd.apache.org/>. Acesso em 30/04/2009.

APACHE LICENSE. **Licença de Distribuição**, 2004**.** Disponível em < http://www.apache.org/licenses/LICENSE-2.0>. Acesso em 30/04/2009.

ARTIST. **ARTIST Control Remote Laboratory.** Disponível em <http://rtaixml.dsi.unifi.it/plants/choice_process/>. Acesso em 30/04/2009.

BASSO, M., BAGNI, G. ARTIST: A Real-Time Simulink-based Telelab. In: PROCEEDINGS OF 2004 IEEE CONFERENCE ON COMPUTER AIDED CONTROL SYSTEMS DESIGN (CACSD), Taipei, Taiwan, 2004.

BASSO, M., BUCHER, R., ROMAGNOLI, M. e VASSALI, M. Real-Time Control with Linux: A Web Service Approach. In: PROCEEDINGS OF 44TH. IEEE CONFERENCE ON DECISION AND CONTROL, AND THE EUROPEAN CONTROL CONFERENCE (CDC-ECC), Sevilla, Espanha, 2005.

BASTOS, V. L. Curso Presencial Ou Curso A Distância? : Aspectos Econômicos Do Processo Decisório. **Linhas Críticas**, Brasília, v. 7, n. 13, p. 265-274, 2001.

BERTOCCO, M.; FERRARIS, F.; OFFELLI, C.; PARVIS, M. A client-server architecture for distributed measurement systems. **IEEE Transaction on Instrumentation and Measurement**, v. 47, n. 5, 1998.

BOOCH, G., RUMBAUGH, J., JACOBSON, I. **The Unified Modeling Language User Guide**. Massachusetts: Editora Addison Wesley, 1998.

BRUYNINCKX, H. **Real Time and Embedded Guide**. Leuven: Katholieke Universiteit Leuven, 2002.

CAMARGOS, L. F. M.; MENEZES, M. A. F. **Introdução a HTML e PHP**. Rio de Janeiro: Ed. Ciência Moderna, 2008.

CAMPBELL, S. L.; CHANCELIER J.; NIKOUKHAH, R. **Modeling and Simulation in Scilab/Scicos.** New York: Springer Science + Business Media, 2006.

CASINI, M., PRATTICHIZZO, D., VICINO, A. The automatic control telelab: a web-based technology for distance learning. **IEEE Control System Magazine**, University of Michigan, v. 24, n. 3, p. 36-44, 2004.

CLARK, R. E. Reconsidering research on learning from media. **Review of Educational Research**, University of Southern California, v. 53, n. 4, p. 445-459, 1983.

CLOUTIER, O.; MANTEGAZZA, P.; PAPACHARALAMBOUS, S.; SOANES, I.; HUGHES, S.; YAGHMOUR, K. DIAPM-RTAI Position Paper**.** In: RTSS 2000 - REAL TIME OPERATING SYSTEMS WORKSHOP, Florida, United States, 2000.

COMEDI. **Linux Control and Measurement Device Interface.** Disponível em <http://www.comedi.org>. Acesso em 30/04/2009.

CONALLEN, J. Modeling Web applications architectures with UML. **Communications of ACM**, v. 42, n. 10, p. 63-70, 1999.

CONALLEN, J. **Desenvolvimento de Aplicações Web com UML**, Rio de Janeiro: Editora Campus, 2003.

COSTAS, R. Já vale a pena fazer pós a distância. **Revista Veja**, São Paulo, ano 39, n. 1943, p. 94-95, 2006.

DOZIO, L; MANTEGAZZA, P. Real Time Distributed Control System using RTAI. In: PROCEEDINGS OF THE SIXTH IEEE INTERNATIONAL SYMPOSIUM ON OBJECT-ORIENTED REAL-TIME DISTRIBUTED COMPUTING (ISORC'03), Hakodate, Japan, 2003.

DZENDZIK, I. T. **Processo de Desenvolvimento de Web Sites com Recursos da UML.** São José dos Campos, 182 p., 2004. Dissertação (Mestrado), Instituto Nacional de Pesquisas Espaciais.

EMERSION. **Remote Lab of the Automatic Control Laboratory.** Disponível em< http://lawww.epfl.ch/page13172.html>. Acesso em 30/04/2009.

E-JOURNALS. **Online Journals**. Disponível em <http://www.e-journals.org/>. Acesso em 30/04/2009.

FARIA, C. G. e MOSCOSO, M.N.A. **Simulação Híbrida de um Sistema Dinâmico do tipo Ball-Beam com a Implementação de um Controlador Digital em Tempo Real.** Belém, 62p., 1997. Trabalho de Conclusão de Curso, Universidade Federal do Pará.

FARINES, J.; FRAGA, J. S.; OLIVEIRA, R. S. **Sistemas de Tempo Real.** Santa Catarina: Universidade Federal de Santa Catarina, 2000.

FORTE, C.; SANTIN, R.; OLIVEIRA, F. C.; KIRNER, C. Implementação de Laboratórios Virtuais em Realidade Aumentada para Educação à Distância. In: V WORKSHOP DE REALIDADE VIRTUAL E AUMENTADA, Bauru. **Anais...** Bauru: Unesp Editora, 2008. v. 1. p. 20-28.

FRANKE, M. **A Quantitative Comparison of Realtime Linux Solutions**. Disponível em <http://rtg.informatik.tu-chemnitz.de/docs/da-sa-txt/sa-franm.pdf>. Acesso em 30/04/2009.

GONÇALVES, E. **Desenvolvendo Aplicações Web com JSP, SERVELTS, JAVASERVER FACES, HIBERNATE, EJB 3, PERSISTANCE E AJAX**. Rio de Janeiro: Ed. Ciência Moderna, 2007.

GOOGLE. **Google Search**. Disponível em <http://www.google.com.br/>. Acesso em 30/04/2009.

HAUSER, J.; SASTRY, S.; KOKOTOVIC, P. Nonlinear control via approximate input-output linearization: the ball and beam example. **IEEE Transaction on Automatic Control,** v. 37, i. 3, p. 392-398, 1992.

HERTZOG, H. M.; BASSO, L. O.; BALLESTER, D. A. P.; SILVEIRA, J. G.; CASTELLO, F. C. Desenvolvimento de um software educacional baseado na internet para estudos de casos clínicos. **Revista Novas Tecnologias na Educação,** Universidade Federal do Rio Grande do Sul, v. 3, n. 2, 2005.

HOON, S. P. Conducting Experiments over the internet. **Instrumentation Newsletter**, National Instruments, v. 10, n. 4, 1998.

IBM. **IBM Rational Rose**. Disponível em <http://www.ibm.com/software/rational/>. Acesso em 30/04/2009.

JANTZEN, J. Design of fuzzy controllers, **Technical University of Denmark: Department of Automation**, Dinamarca, Technical Report n. 98-E-864, 27 p., 1999.

JETTY. **Jetty Webserver.** Disponível em <http://www.mortbay.org/>. Acesso em 30/04/2009.

JFORUM. **JForum - database-independent and multi-threaded forum software.** Disponível em <http://www.jforum.net>. Acesso em 30/04/2009.

KUROSE, J. F.; ROSS, K. W. **Redes de computadores e a Internet**. São Paulo: Editora Pearson Education do Brasil Ltda, 2006, ed. 3.

LABVIEW. **The software that powers virtual instrumentation.** Disponível em <http://www.ni.com/labview>. Disponível em 30/04/2009.

LAURENT, S. S.; JOHNSON, J.; DUMBILL, E. **Programming Web Services with XML-RPC.** Cambridge: Editora O'Reilly, 2001.

LYNXOS. **Real Time Operating System from LynuxWorks.** Disponível em <http://www.lynuxworks.com/>. Acesso em 30/042009.

MARCELO, A. **Apache:** Configurando o servidor WEB para Linux. Rio de Janeiro: Ed. Brasport, 2006.

MAGRABI, F.; LOVELL, N. H.; CELLER, B. G. A Web-Based approach for electrocardiogram monitoring in the home. **International Journal of Medical Informatics**, Elsevier Ireland, vol. 54, n. 2, p. 145-153, 1999.

MATLAB. **The Mathworks – MATLAB and Simulink for Technical Computing**. Disponível em <http://www.mathworks.com>. Acesso em 30/04/2009.

MATSUMOTO, E. Y. **Matlab 6.5: Fundamentos de programação**. São Paulo: Editora Érica, 2004.

MERLOT. **Multimedia Educational Resource for Learning and Online Teaching**. Disponível em <http://www.merlot.org/merlot/index.htm>. Acesso em 30/04/2009.

MICHIGAN. **University of Michigan**: Control Tutorials for Matlab and Simulink. Disponível em <http://www.engin.umich.edu/class/ctms/simulink/examples/motor2/motor2s.htm>. Acesso em 30/04/2009.

MICROSOFT IIS. **Internet Information Service.** Disponível em <http://www.microsoft.com/windowsserver2003/iis/default.mspx>. Acesso em 30/04/2009.

MOCONET. **Control Engineering Laboratory.** Disponível em <http://www.control.hut.fi/dev/MoCoNet/>. Acesso em 30/04/2009.

MOODIE, M. **Pro Apache Tomcat 6**. New York: Editora Apress, 2007.

MOODLE. **Open Source Community-based tools for learning.** Disponível em <http://moodle.org/>. Acesso em 30/04/2009.

MOSSIN, E. A. **Laboratório remoto para ensino a distância de sistemas de controle distribuído.** São Carlos, 168 p., 2007. Dissertação (Mestrado) – Universidade de São Paulo.

NASCIMENTO, A. C. A. **Princípios de design na elaboração de material multimídia para a Web**. Projeto Rede Internacional Virtual de Educação (RIVED), Ministério da Educação, Brasília, 2005. Disponível em <http://www.rived.mec.gov.br/artigos/multimidia.pdf>. Acesso em: 30/04/2009.

NETCRAFT, 2009. **Web Server Survey**. Disponível em <http://news.netcraft.com/>. Acesso em 30/04/2009.

NETO, A. R.; TERUEL, B. Supervisão e controle automático de sistemas de secagem de produtos agrícolas através do software TERMICONT. **Revista Brasileira de Engenharia de Biossistemas**, Campinas, v. 2, n. 1, p. 071-079, 2008.

NGUYEN-NGOC, A. V.; REKIK, Y.; GILLET, D. A framework for sustaining the continuity of interaction in Web-based learning environment for engineering education. In: PROCEEDINGS OF THE WORLD CONFERENCE ON EDUCATIONAL MULTIMEDIA, HYPERMEDIA AND TELECOMMUNICATIONS (ED-MEDIA 2005), (C) AACE, Montreal, Canada, 2005.

NISE, N. S. **Engenharia de Sistemas de Controle**. São Paulo: Editora Santuário, 2000.

NISKIER, A. **Educação à Distância:** A Tecnologia da Esperança**.** São Paulo: Edições Loyola, 1999.

OGATA, K. **Engenharia de Controle Moderno**. Rio de Janeiro: Ed. Prentice Hall do Brasil, 1985.

PALOP, J. M. G.; TERUEL, J. M. A. Virtual work bench electronic instrumentation teaching. **IEEE Transaction on Education**, v. 43, n. 1, 2000.

PARASKEVOPOULOS, P. B. **Digital Control Systems**. London: Ed. Prentice Hall International, 1996.

PLOTEGUER, S. L.; FERNANDES, M. M. Raciocínio Baseado em Casos Aplicado a um Sistema de Diagnóstico Remoto. In: XXV CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO (SBC), São Leopoldo, Brasil, 2005.

POHJOLA, M. **PID Controller Design in Networked Control Systems.** Helsinki, 82 p., 2006. Dissertação (Mestrado) – Universidade de Helsinki.

RACCIU, G.; MANTEGAZZA, P. **RTAI 3.4 User Manual rev. 0.3**. Documento de referência oficial, 2006. Disponível em <https://www.**rtai**.org/index.php?module=documents&JAS_DocumentManager_op=downloadFile&JAS_File_id=46>. Acesso em 30/04/2009.

ROTHE-NEVES, M**. Construção de protótipo para laboratório de ensino e pesquisa: sistema ball-beam.** Belém, 81 p., 1999. Trabalho de Conclusão de Curso, Universidade Federal do Pará.

RTAI-XML. **RTAI's Server Component.** Disponível em <http://artist.dsi.unifi.it/rtaixml/>. Acesso em 30/04/2009.

RTLINUXPRO. **FSMLabs RTLinuxPro.** Disponível em **<**http://www.diamondsystems.com/products/rtlinuxpro>. Acesso em 30/04/2009.

SANKARAYOGI, R. **Software Tools for Real-Time Simulation and Control.** West Virginia, 83 p., 2005. Dissertação (Mestrado) - West Virginia University.

SCHUMACHER, E., SILVA, H. S., SILVA, M. R., DALFOVO, O. Experiências Virtuais aplicadas em Aulas de Teoria de Física. In: WORKCOMP SUL, Florianópolis. **Anais…** Florianópolis: UNISUL, 2004.

SCILAB. **The open source platform for numerical computation.** Disponível em <http://www.scilab.org>. Acesso em 30/04/2009.

SECOND LIFE. **Virtual Worlds, avatars, 3d chats, online meetings.** Disponível em **<**http://www.secondlife.com>. Acesso em 30/04/2009.

SILVA, G. M. **Guia Foca GNU/Linux.** Disponível em <http://www.guiafoca.org>. Acesso em 30/04/2009.

SILVA, O. F. **Mídias e tecnologias instrucionais para o ensino/aprendizado de sistemas de controle.** Florianópolis, 118 p., 2001. Tese (Doutorado) – Universidade Federal de Santa Catarina.

SIMMONS, D. E. The forum report: E-learning adoption rates and barriers. In: ROSSET, A. (Ed.), **The ASTD e-learning handbook**. New York: McGraw-Hill, 2002, p. 19-23.

STARUML. **The Open Source UML/MDA Platform**. Disponível em <http://staruml.sourceforge.net/en>. Acesso em 30/04/2009.

STEGAWSKI, M. A.; SCHAUMANN, R. A new virtual-instrumentation-based experimenting environment for undergraduate laboratories with application in research and manufacturing. **IEEE Transaction on Instrumentation and Measurement**, v. 47, n. 6, 1998.

SUGENO, M.; KANG, G., T. Structure identification of fuzzy model. **Fuzzy Sets and Systems**, v. 28, p. 15-33, 1988.

SUN. **Sun Microsystems.** Disponível em <http://www.sun.com>. 30/04/2009.

TAKAGI, T.; SUGENO, M. Fuzzy identification of systems and its applications to modeling control. **IEEE Transactions on Systems, Man and Cybern**, v. 15, n. 1, p. 116-132, 1985.

TANENBAUM, A. S. **Redes de computadores**. Rio de Janeiro: Editora Campus, 2003, ed. 4.

TANENBAUM, A. S; STEEN, M. V. **Sistemas Distribuídos – Princípios e Paradigmas**. Rio de Janeiro: Ed. Prentice Hall do Brasil, 2007, ed. 2.

TELEDUC. **Ensino à Distância.** Disponível em <http://www.teleduc.org.br>. Acesso em 30/04/2009.

TOMCAT. **An Open Source JSP and Servlet Container from the Apache Foundation**. Disponível em <http://tomcat.apache.org/>. Acesso em 30/04/2009.

TORRES, G.; LIMA, C. **Como o Protocolo TCP/IP Funciona**. Tutorial Clube do Hardware, 2007. Disponível em **<**http://www.clubedohardware.com.br/artigos/1351/>. Acesso em 30/04/2009.

ULINUX. **Lineo's Embedded Linux**. Disponível em <http://www.lineo.co.jp/eng/products-services/products/ulinux.html>. Acesso em 30/04/2009.

WANG, L. X. **A Course in Fuzzy Systems and Control**, London: Ed. Prentice Hall International, 1997.

WENGER, E., MCDERMOTT, R., & SNYDER, W. **Cultivating communities of practice: A guide to managing knowledge**. Cambridge: Harvard Business School Press, 2002.

WISINTAINER, M. A. **RexLab: Laboratório de Experimentação Remota com o Microcontrolador 8051**. Florianópolis, 137 p., 1999. Dissertação (Mestrado) – Universidade Federal de Santa Catarina.

WIKIPEDIA. **A Enciclopédia Livre.** Disponível em <http://www.wikipedia.org/>. Acesso em 30/04/2009.

XENOMAI. **Xenomai: Real-Time Framework for Linux.** Disponível em <http://www.xenomai.org/index.php/Main_Page>. Acesso em 30/04/2009.

XML-RPC. **Cross-platform distributed computing, based on the standards of the internet.** Disponível em <http://www.xmlrpc.com/>. Acesso em 30/04/2009.

APÊNDICE A – SISTEMAS DE CONTROLE

Um sistema de controle consiste em um subsistema ou uma série de subsistemas que podem gerenciar, controlar, dirigir ou regular o comportamento de outros sistemas. Sistemas de controle podem ser encontrados em numerosas aplicações, servomotores conectados a uma carga ou foguetes são exemplos. Os sistemas de controle podem ser configurados em malha aberta ou malha fechada. Sistemas de controle em malha aberta são sistemas em que a saída não tem efeito na ação de controle, ou seja, a saída não é medida nem realimentada para comparação com a entrada, sendo assim, este tipo de sistema não corrige efeitos produzidos por perturbações e são comandados unicamente com base na entrada (NISE, 2000). Este tipo de sistema pode observado na figura 48.

Figura 48 Diagrama de Blocos de um sistema de controle em malha aberta

Em sistemas de controle em malha fechada a saída possui um efeito direto na ação de controle, ou seja, são sistemas realimentados. O sinal de erro, diferença entre o sinal de entrada e o sinal realimentado, é alimentado no controlador de modo a reduzir o erro e manter a saída do sistema em um valor desejado (OGATA, 1985). A figura 49 representa este tipo de sistema. O bloco de soma algébrica fecha a malha, ou seja, representa a realimentação do sistema, gerando o sinal de erro. A partir do sinal de erro, o bloco Controlador gera o sinal de controle a ser aplicado na Planta ou Processo.

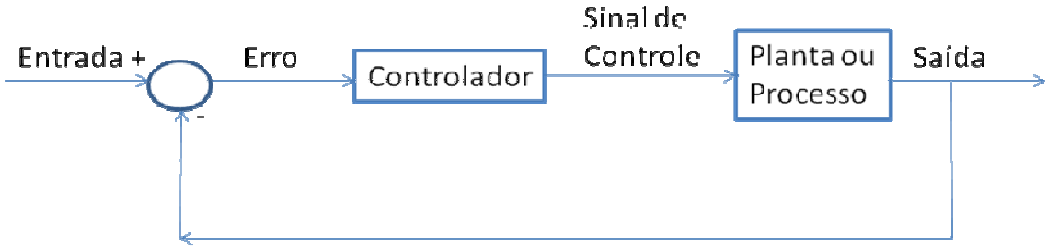

Figura 49 Diagrama de Blocos de um sistema de controle em malha fechada

Geralmente o controlador deverá processar o sinal de erro seguindo três requisitos, que integram o projeto de sistemas de controle: resposta transitória, estabilidade e erro de regime permanente (NISE, 2000). Para obedecer estes requisitos deve-se desenvolver um modelo para a planta, projetar um controlador baseado no modelo desenvolvido e no critério de controle (especificação), simulações digitais para validação do modelo e testes no sistema real. Existem diversas estruturas de controle existentes, entretanto serão demonstrados neste trabalho o controlador Proporcional, Integral e Derivativo (PID), o controlador Fuzzy e a realimentação de estados.

1 CONTROLADOR PID

O controlador PID é constituído pela combinação da ação de controle proporcional, integral mais derivativa, combinando as vantagens individuais de cada uma destas ações de controle. A ação de controle proporcional, na verdade, é um amplificador com um ganho ajustável (OGATA, 1985). Este ganho é representado pela seguinte equação, em termos de função de transferência (A.1):

$$\frac{M(s)}{E(s)} = Kp \tag{A.1}$$

Onde *M(s)* representa a saída do controlador, *E(s)* o sinal de erro e Kp a sensibilidade proporcional ou ganho.

A ação de controle integral consiste em aplicar um sinal de controle proporcional à integral do sinal de erro, ou seja, se *E(s)* dobrar de valor, então *M(s)* varia duas vezes mais rápido e quando *E(s)* for nulo, *M(s)* permanece estacionário, garantindo, assim, o rastreamento de uma referência com erro nulo em regime permanente. A função de transferência (A.2) representa a ação de controle proporcional mais integral:

$$\frac{M(s)}{E(s)} = Kp\left(1 + \frac{1}{T_i s}\right) \tag{A.2}$$

Onde T_i representa o tempo integral. Os parâmetros Kp e T_i são ajustáveis. O tempo integral ajusta a ação de controle integral, enquanto uma mudança no valor de Kp afeta tanto a parte proporcional quanto a parte integral da ação de controle.

A ação de controle derivativa consiste na aplicação de um sinal de controle proporcional a derivada do sinal de erro. A derivada de uma função esta relacionada intuitivamente com a tendência de variação desta função em um determinado instante de tempo. Assim, aplicar como controle um sinal proporcional à derivada do sinal de erro é equivalente a aplicar uma ação baseada na tendência de evolução do erro. A ação derivativa é então dita *antecipatória* ou *preditiva* e tende a fazer com que o sistema reaja mais rapidamente (OGATA, 1985). Este fato faz com que a ação derivativa seja utilizada para a obtenção de respostas transitórias mais rápidas, ou seja, para a melhora do comportamento dinâmico do sistema em malha fechada. Ratifica-se que a ação de controle derivativa não pode ser utilizada sozinha, pois esta ação somente é efetiva durante os intervalos de tempo correspondentes a transitórios. Logo, a função de transferência (A.3) representa a ação de controle PID:

$$\frac{M(s)}{E(s)} = Kp\left(1 + \frac{1}{T_i s} + T_d s\right) \tag{A.3}$$

Onde T_d representa o tempo derivativo. Assim, a ação de controle proporcional deve reduzir o tempo de subida e reduzir, mas não eliminar o erro de regime permanente. A ação integral deve eliminar o erro de regime permanente, mas pode piorar a resposta transitória. A ação derivativa deve aumentar a estabilidade do sistema, reduzir o overshoot e melhorar a resposta transitória. O diagrama de blocos do controlador PID pode ser observado na figura 50:

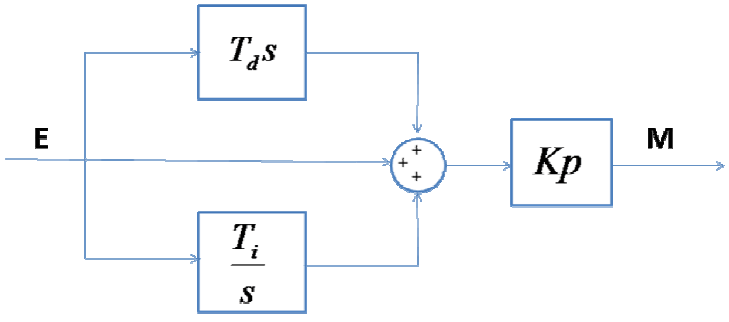

Figura 50 Controlador PID

Os valores apropriados para *Kp, Ti* e *Td* podem ser determinados pelo método da resposta transiente, proposto por Ziegler e Nichols (PARASKEVOPOULOS, 1996). Neste método o sistema será excitado por uma função degrau unitário apresentando um formato de resposta transiente, em malha aberta, semelhante à figura 51.

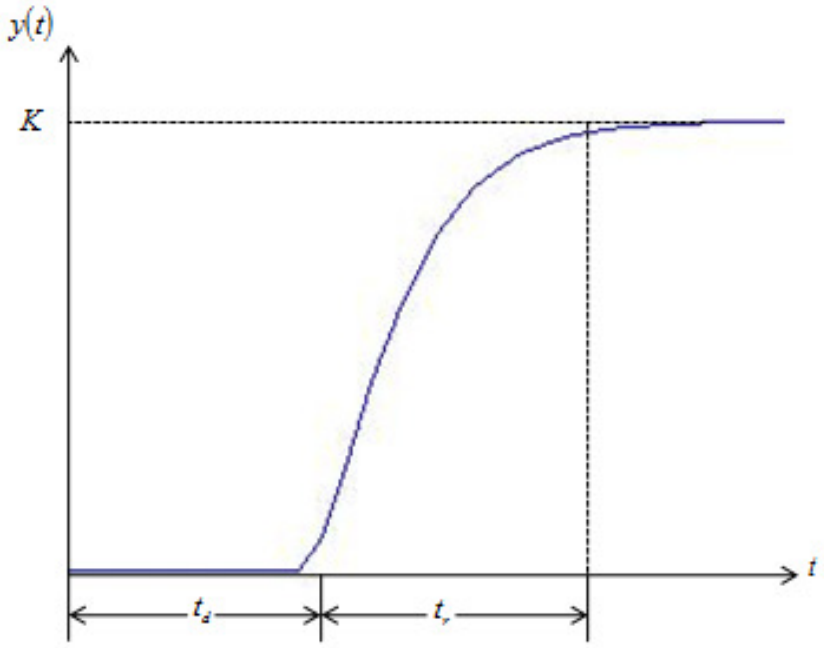

Figura 51 Resposta do sistema a uma função degrau

São introduzidas as variáveis θ e t_d, onde $\theta = \frac{K}{t_r}$ = tempo de reação e t_d = atraso. Para atingir um fator de amortecimento próximo de 0.2, os valores de *Kp, Ti* e *Td* podem ser selecionados conforme a tabela 4.

Tabela 4

Possíveis valores para Kp, Ti e Td

Controlador		Kp	Ti	Td
Proporcional	P	$\frac{1}{t_d\theta}$		
Proporcional - Integral	PI	$\frac{0.9}{t_d\theta}$	$3t_d$	
Proporcional -Integral – Derivativo	PID	$\frac{1.2}{t_d\theta}$	$2t_d$	$0.5t_d$

Onde:

$$K_i = \frac{K_p}{T_i} \tag{A.4}$$

$$K_d = K_p * T_d \tag{A.5}$$

Resultando no diagrama de blocos apresentado na figura 52:

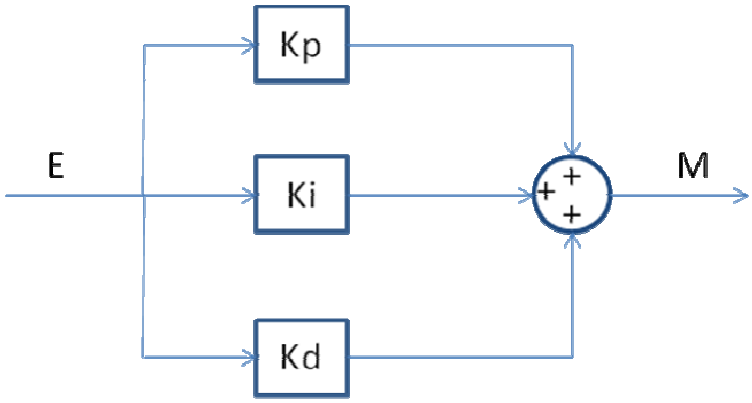

Figura 52 Controlador PID

2 CONTROLADOR FUZZY

Sistemas fuzzy são sistemas baseados em conhecimento ou baseados em regras, que podem ser utilizados para traduzir em termos matemáticos a informação imprecisa expressa por um conjunto de regas lingüísticas. São sistemas onde o conhecimento é expresso através de regras SE-ENTÃO, ou seja, se um operador humano for capaz de articular sua estratégia de ação como um conjunto de regras SE-ENTÃO, logo um algoritmo passível de ser implementado em computador pode ser construído. O resultado será um sistema de inferência baseado em regras, no qual a teoria de sistemas fuzzy fornece o ferramental matemático necessário para lidar com tais regras (WANG, 1997). Por exemplo, a sentença abaixo é um exemplo de regra fuzzy SE-ENTÃO:

SE a velocidade do carro é alta ENTÃO aplicar menos força no acelerador (1.1)

Onde as palavras “alto” e “menos” são caracterizadas por funções de pertinência. Por sua vez, um controlador fuzzy é uma técnica de controle baseada em informações heurísticas sobre o processo. A técnica de controle fuzzy fornece uma metodologia formal para se representar, manipular e implementar o conhecimento humano heurístico sobre como

controlar um sistema (JANTZEN, 1999). A principal característica do controle fuzzy é que ele trabalha com regras lingüísticas ao invés de modelos matemáticos e relacionamentos através de funções. Utilizando técnicas de controle convencionais, as decisões tomadas pelos controladores são rigidamente verdadeiras ou falsas. O controle fuzzy utiliza a lógica fuzzy, que está mais próxima do modo humano de pensar e se expressar, comparado com sistemas de controle convencionais (PARASKEVOPOULOS, 1996).

Basicamente, sistemas fuzzy podem ser classificados de dois modos: Sistemas Fuzzy Takagi-Sugeno-Kang (TSK) e Sistemas Fuzzy Mandani. Em sistemas TSK as entradas e saídas do sistema são variáveis que utilizam valores reais (TAKAGI; SUGENO, 1985) e (SUGENO; KANG, 1988). Ao invés de utilizar regras fuzzy na forma de (1.1), o sistema TSK utiliza regras da seguinte forma:

SE a velocidade x do carro é alta ENTÃO a força no acelerador é y=cx (1.2)

Onde a palavra "alta" tem o mesmo significado de (1.1), e a letra "c" é uma constante. Comparando (1.1) e (1.2) pode-se observar que a parte ENTÃO das regras muda de uma descrição utilizando palavras para uma simples fórmula matemática. Esta mudança facilita a combinação de regras. De fato, o sistema TSK é uma média ponderada dos valores da parte ENTÃO das regras (WANG, 1997). A figura 53 apresenta a configuração básica do sistema TSK:

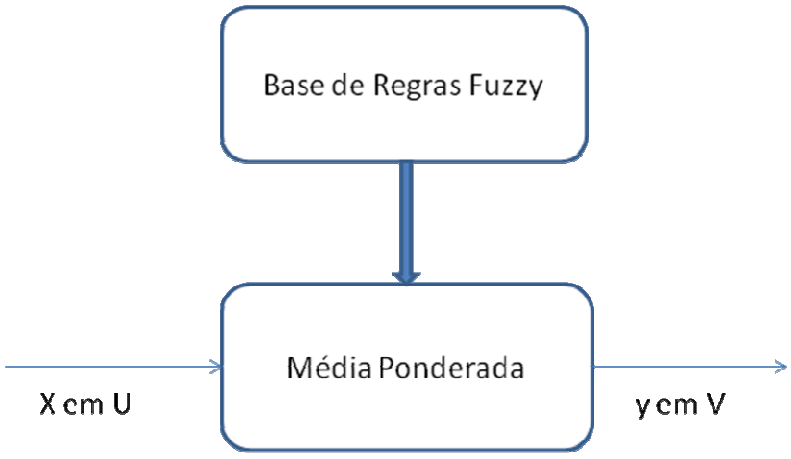

Figura 53 Configuração Básica de um sistema fuzzy TSK

De modo geral um sistema fuzzy Mandani é composto por 4 partes: fuzzificador, base de regras, máquina de inferência e defuzzificador, conforme figura 54.

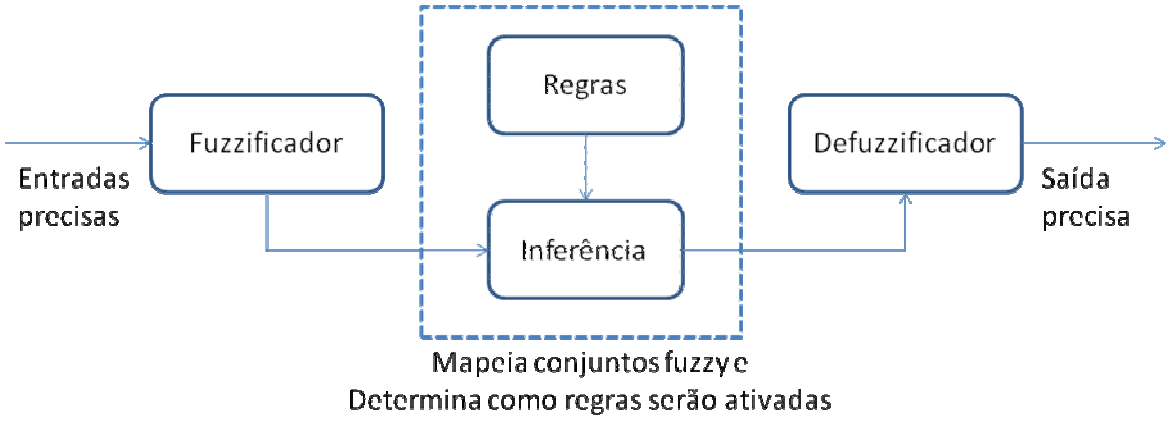

Figura 54 Configuração sistema fuzzy Mandani

Em sistemas fuzzy Mandani considera-se a utilização de entradas precisas, ou não-fuzzy, resultantes de medições e observações. Sendo assim é necessário realizar a conversão do valor real da entrada para conjuntos fuzzy relevantes, correspondendo ao estágio de fuzzificação. Nesta fase são ativadas também as regras relevantes para uma determinada situação. A base de regras é fornecida por especialistas, na forma de sentenças do tipo SE-ENTÃO, entretanto esta pode não ser uma tarefa fácil, por isso existem métodos alternativos de extração de regras de dados numéricos.

Na fase de inferência ocorrem as operações com conjuntos fuzzy propriamente ditas: combinação dos antecedentes com os conseqüentes através dos mecanismos de inferência *modus ponens* ou *modus tollens* (WANG, 1997). Os conjuntos fuzzy de entrada, relativos aos antecedentes das regras, e o de saída, referente ao conseqüente, podem ser definidos previamente ou, alternativamente, gerados automaticamente a partir dos dados. A fase final do sistema fuzzy, a defuzzificação, é onde os conjuntos fuzzy são convertidos em valores reais.

O bom desempenho de controladores fuzzy depende também da definição das variáveis de entrada (antecedente) e das variáveis de saída (conseqüente), pois será afetado pelo número de conjuntos e sua forma (triangular, trapezoidal ou elíptica). A sintonia das funções de pertinência destas variáveis pode ser realizada de forma manual ou automática. A figura 55 representa a definição das funções de pertinência para a variável velocidade, por exemplo.

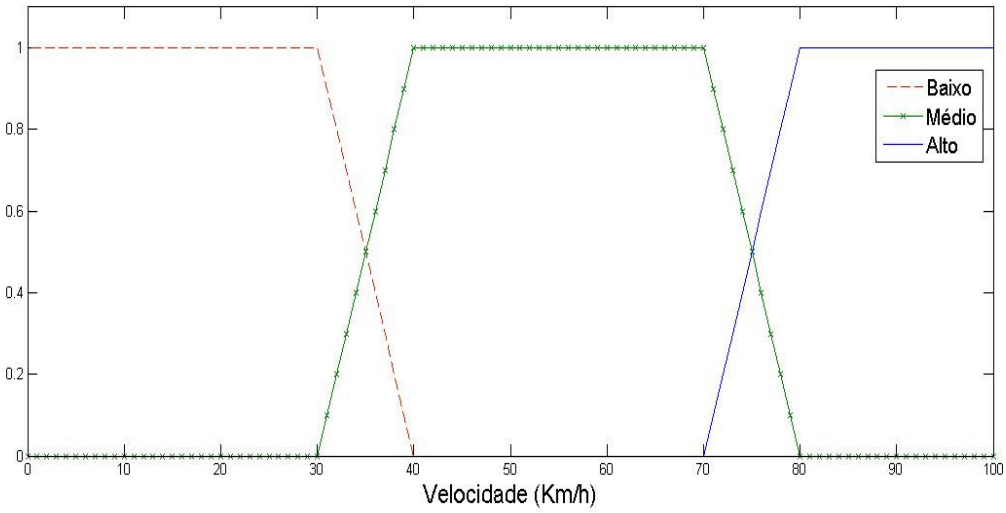

Figura 55 Funções de Pertinência para a variável Velocidade

3 REALIMENTAÇÃO DE ESTADOS

Enquanto a teoria de controle convencional baseia-se na relação entre entrada e saída, ou função de transferência, a teoria de controle moderno baseia-se na descrição das equações do sistema em termos de *n* equações diferenciais de 1ª ordem que podem ser combinadas em uma equação diferencial de 1ª ordem na forma matricial. O uso de notação matricial simplifica em muito a representação matemática de sistemas de equações. O aumento do número de variáveis de estado, o número de entradas e saídas não aumenta a complexidade das equações (OGATA, 1985).

Desta forma, para se obter um controlador utilizando realimentação de estados, o processo a ser controlado deve ser representado por suas equações de estado e de saída, conforme demonstrado na equação (A.6), para o caso linear e invariante no tempo.

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned} \tag{A.6}$$

Se o vetor de estados, $x(t)$ é totalmente disponível, ou seja, se todas as variáveis de estado podem ser medidas (ou pelo menos estimadas), realimenta-se cada uma das variáveis

de estado devidamente ponderadas por ganhos (valores reais), para gerar um controlador por realimentação de estados.

Dependendo da natureza do sinal de referência, duas estruturas são definidas para este controlador: sendo nula a referência tem-se o problema de regulação e sendo a referência um sinal dependente do tempo, tem-se o problema do rastreamento (SILVA, 2001).

Para a regulação pode-se propor como ação de controle a equação (A.7):

$$u = -K.x = -\begin{bmatrix} K_1 & K_2 & K_3 & K_4 \end{bmatrix}.\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \tag{A.7}$$

O diagrama de blocos do sistema realimentado, para o problema regulatório, é mostrado na Figura 56.

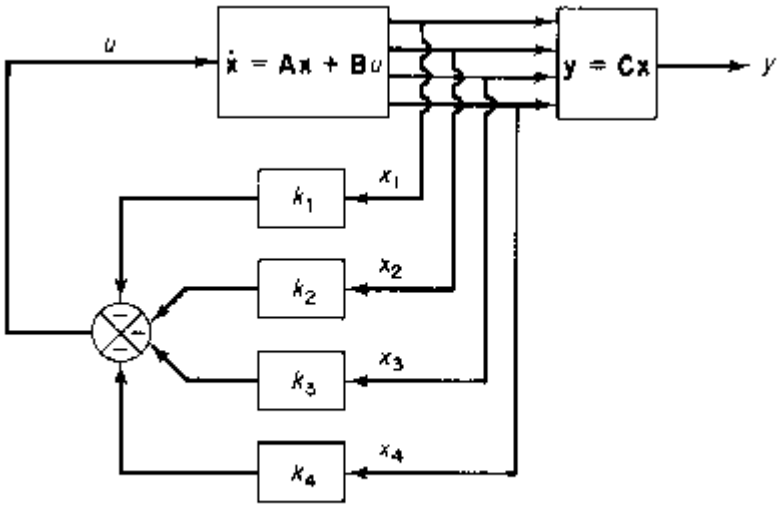

Figura 56 Diagrama de Bloco para o problema de regulação

Para o problema de rastreamento, onde a saída *y*, igual a variável de estado *x1*, deve acompanhar o sinal de referência variante no tempo *r(t)*, injetado na entrada do sistema, pode-se propor como ação de controle a equação (A.8)

.

$$u = -K.x + K_1(r - y) = -\begin{bmatrix} K_1 & K_2 & K_3 & K_4 \end{bmatrix}.\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + K_1(r - Cx) \qquad \text{(A.8)}$$

O diagrama de blocos do sistema realimentado é mostrado na Figura 57.

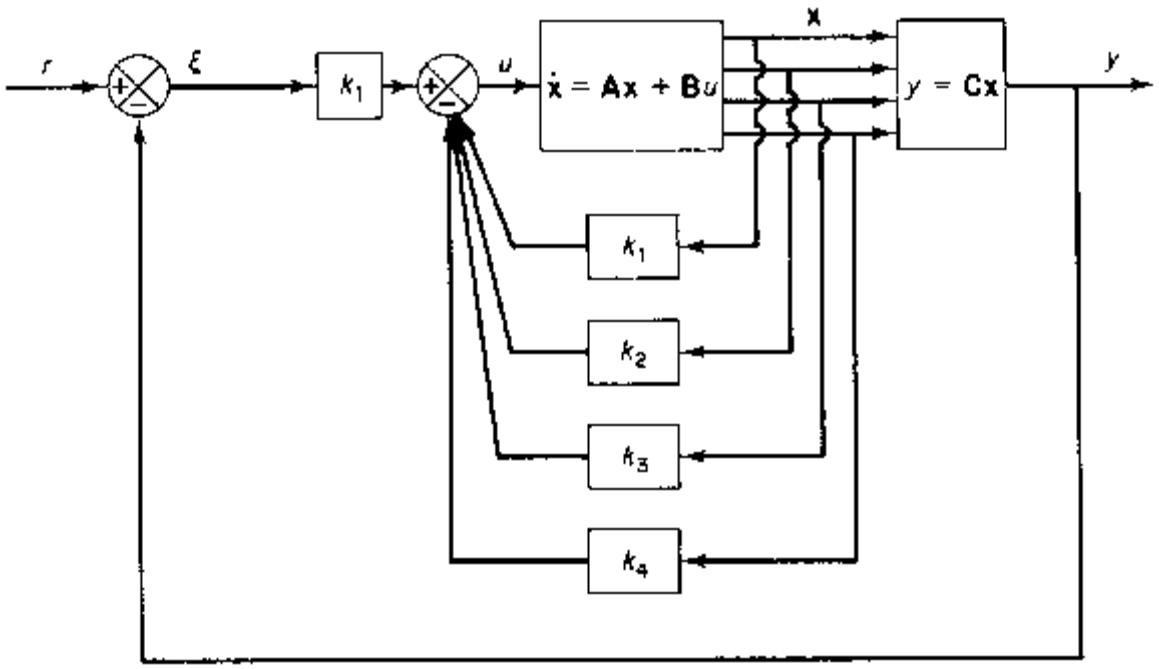

Figura 57 Diagrama de Blocos para o problema de rastreamento

APÊNDICE B – MODELOS MATEMÁTICOS DE SISTEMAS FÍSICOS

As maiorias dos sistemas dinâmicos reais, independente de serem mecânicos, elétricos, térmicos, hidráulicos, econômicos ou biológicos, podem ser descritos matematicamente por equações diferenciais. Deste modo, a resposta de um sistema dinâmico real quando sujeito a uma dada entrada (ou função de excitação), pode ser obtida a partir da solução das equações diferenciais que o descrevem (OGATA, 1995). A figura 58 apresenta a representação de sistemas físicos.

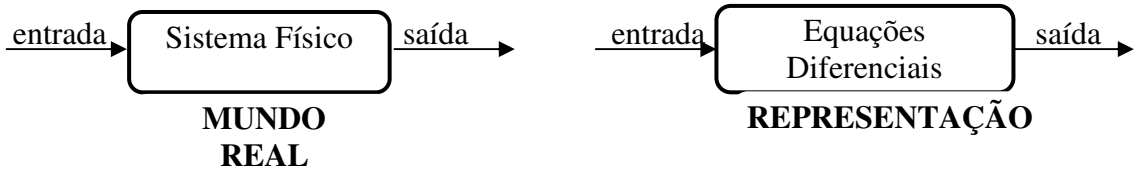

Figura 58 Representação de sistemas físicos

As equações diferenciais que descrevem as características dinâmicas de um dado sistema físico são também denominadas modelo matemático, e podem ser obtidas a partir das leis físicas que governam um sistema em particular, como exemplo, as lei de Newton para sistemas mecânicos e as de Kirchhoff para sistemas elétricos.

O primeiro passo na análise de um sistema dinâmico é obter seu modelo. Na obtenção de um modelo matemático deve-se estabelecer o compromisso entre simplicidade do modelo e precisão dos resultados por ele gerados. Em geral, na solução de um novo problema, deve-se construir, inicialmente, um modelo simplificado de modo a ganharmos um conhecimento básico e geral para a solução. Posteriormente um modelo matemático mais completo poderá então ser desenvolvido e utilizado para uma análise mais integral (OGATA, 1985).

Uma classe de sistemas dinâmicos que apresenta modelos matemáticos simples são os lineares e invariantes no tempo. O modelo para tais sistemas se constitui em equações diferenciais lineares a parâmetros constantes. A seguir será apresentada a modelagem matemática de um circuito L-R (SILVA, 2001).

Um Circuito L-R é constituído de um indutor com indutância L, em série com um resistor de resistência R, sobre os quais pode-se aplicar uma tensão *ei(t)* provocando uma corrente *i(t)* e quedas de tensão $e_l(t)$ e $e_r(t)$ no indutor e resistor respectivamente, conforme é mostrado na Figura 59.

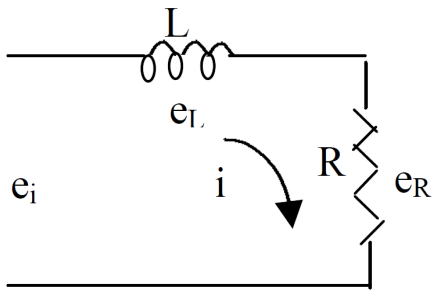

Figura 59 Circuito L-R

A lei de Kircchoff para quedas de tensão é dada pela equação (B.1).

$$\sum V_i(t) = 0 \tag{B.1}$$

Ou seja, a soma algébrica das quedas de tensão num circuito fechado é igual a zero, então, para o circuito da Figura 59, a equação (B.1) resulta na equação (B.2).

$$e_i(t) - e_r(t) - e_l(t) = 0 \tag{B.2}$$

E como as tensões no resistor e indutor são dadas pela equação (B.3).

$$\begin{aligned} e_r(t) &= Ri(t) \\ e_l(t) &= L\frac{di(t)}{dt} \end{aligned} \tag{B.3}$$

E o modelo final é mostrado na equação (B.4).

$$e_i(t) = L\frac{di(t)}{dt} + Ri(t) \tag{B.4}$$

Se a saída, a ser observada, for a tensão no resistor $e_r(t)$, e não a corrente de circulação $i(t)$, então o modelo resultante é o da equação (B.5).

$$e_i(t) = \frac{L}{R}\frac{de_r(t)}{dt} + e_r(t) \tag{B.5}$$

E se for a tensão no indutor, então, o modelo é dado pela equação (B.6).

$$e_i(t) = e_l(t) + \frac{L}{R}\int e_l(t) + dt \tag{B.6}$$

APÊNDICE C – IDENTIFICAÇÃO DE SISTEMAS

Enquanto a modelagem matemática de sistemas físicos envolve o desenvolvimento de um modelo baseado nas leis que regem os fenômenos do sistema real, a identificação de sistemas é um procedimento alternativo. Supondo a existência de um sinal de entrada *u(k)* e de saída *y(k)*, de um sistema real qualquer, a identificação de sistemas se propõe a obter um modelo matemático que explique, pelo menos em parte e de forma aproximada, a relação de causa e efeito presente nos dados (AGUIRRE, 2004). Ou seja, a identificação de sistemas, consiste no processo de construção de um modelo e estimação dos melhores valores de parâmetros desconhecidos, a partir de dados experimentais, sendo estes dados obtidos basicamente pela aplicação de um sinal de entrada pré-definido no sistema a ser identificado, e o registro de sua resposta. A partir dessa massa de dados (entrada - resposta) e do estabelecimento de um modelo, procura-se determinar os parâmetros do modelo, baseando-se em algum método (OGATA, 1985).

Dentre os métodos possíveis considera-se um extremamente simples que consiste na análise de resposta em regime permanente e transitória. Tal método não é de uso geral, aplicando-se principalmente para sistemas onde se tem conhecimento de um modelo a partir de leis físicas e para o qual deseja-se estimar os parâmetros (SILVA, 2001). Considerando-se como exemplo um servomotor representado no diagrama de blocos da Figura 60.

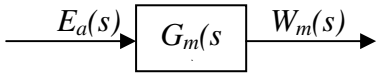

Figura 60 Servomotor

Onde a função de transferência do motor é dada na equação (C.1).

$$G_m(s) = \frac{K}{Ts+1} = \frac{W_m}{E_a} \tag{C.1}$$

Sendo:

E_a → a tensão aplicada na armadura do motor [V];
W → a velocidade angular no eixo do motor [rad/s];
K → ganho do motor;

T → constante de tempo do motor.

A análise de regime permanente permite a determinação do ganho K do motor. O procedimento consiste na aplicação de uma tensão Ea, de magnitude E (constante) no motor. Em regime, o motor atingirá uma velocidade W_m de magnitude W (constante). Pelo teorema do valor final, "s" tende a zero na equação (C.1), resultando a equação (C.2).

$$K = \frac{W}{E} \tag{C.2}$$

Ou seja, o ganho do motor passa a ser simplesmente o quociente entre a magnitude do sinal de entrada e a magnitude da resposta.

A análise da resposta transitória permite a determinação da constante de tempo T do motor. O procedimento consiste no registro da resposta transitória quando da aplicação de uma tensão de magnitude constante (E). Neste caso, utiliza-se a seguinte definição de constante de tempo: o intervalo de tempo necessário para a resposta atingir 63% do seu valor de regime. Dado que é conhecido o valor de regime (e_{ss}), pode-se calcular o ponto da curva de resposta transitória correspondente a 63% deste valor e o instante de tempo em que ocorre, conseqüentemente tem-se a constante de tempo, como é ilustrado na Figura 61..

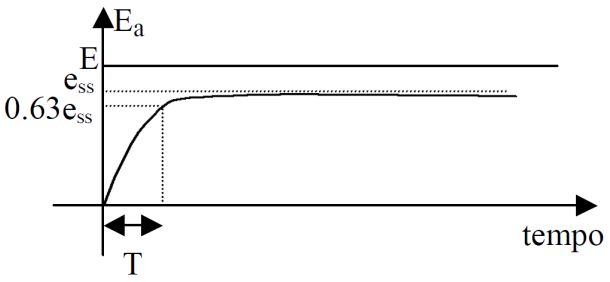

Figura 61 Constante de Tempo
Fonte: (SILVA, 2001)

APÊNDICE D – EXPERIMENTOS DISPONÍVEIS

Atualmente o LabExp disponibiliza aos usuários 3 simulações: o sistema barra-bola; o sistema massa, mola e amortecedor; e o controle de posição de um motor DC.

1 SISTEMA BARRA-BOLA

O sistema barra-bola é um dos modelos mais populares e importantes no ensino de sistemas de controle. Este sistema é bastante utilizado porque é de fácil compreensão, por permitir o estudo de diversas técnicas de controle, tanto clássicas quanto modernas, e por sua instabilidade em malha aberta.

A ação de controle neste sistema é regular automaticamente a posição da bola na barra através da alteração do ângulo desta barra. Este tarefa de regulação torna-se difícil considerando que a bola não permanece somente em uma posição na barra, mas movimenta-se com aceleração proporcional a inclinação da barra. A dinâmica deste sistema é ilustrada na figura 62.

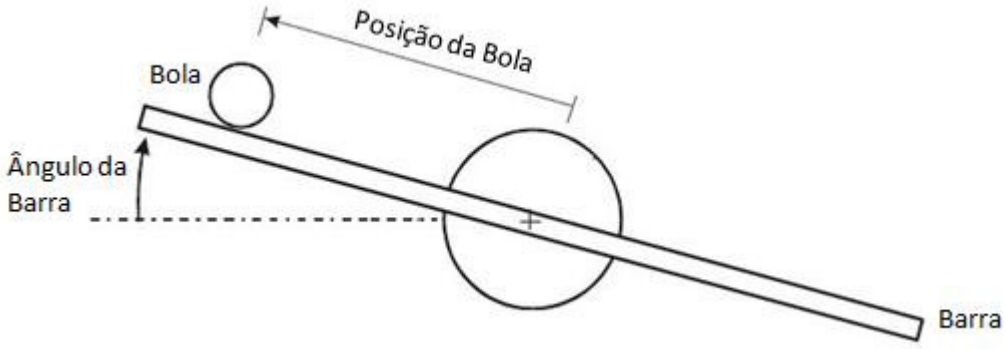

Figura 62 Sistema Barra-Bola

Esta dinâmica é representada através da equação de Lagrange do movimento, de acordo com (HAUSER et al., 1992) e (FARIA; MOSCOSO, 1997), podendo ser modelada segundo as equações abaixo.

$$0 = \left(\frac{Jb}{Rb^2} + mb \right) \ddot{r} + mbg.sen\theta - mbr\ddot{\theta}^2 \quad \text{(D.1)}$$

$$T = \left(mbr^2 + J + Jb \right) \ddot{\theta} + 2mbr\dot{r}\dot{\theta} + mbgr.\cos\theta \quad \text{(D.2)}$$

Onde:

r → deslocamento linear da bola [m];

$\dot{r}$ → velocidade linear da bola [m/s];

$\ddot{r}$ → aceleração linear da bola [m/s²];

θ → deslocamento angular da barra [rad];

$\dot{\theta}$ → velocidade angular da barra [rad/s];

$\ddot{\theta}$ → aceleração angular da barra [rad/s²];

T → torque de entrada [N.m];

g → aceleração de gravidade [m/s²];

J → momento de inércia da barra [kg.m²];

Jb → momento de inércia da bola [kg.m²];

Rb → raio da bola [m];

mb → massa da bola [kg];

Estas equações caracterizam o movimento como não linear. Em projetos de sistemas de controle com características não lineares, múltiplas entradas e múltiplas saídas, a teoria mais adequada é a de controle moderno, que utiliza uma abordagem baseada no conceito de estado (OGATA, 1985). A técnica para o controle no espaço de estados consiste em linearizar o modelo em torno de um ponto de operação e, em seguida, aplicar o método considerando-o linear. A obtenção do modelo e os valores numéricos dos parâmetros foram determinados de acordo com (ROTHE-NEVES, 1999):

$J = 0{,}0059\ kg.m^2;$

$Jb = 4{,}1336.10^{-7}\ kg.m^2;$

$Rb = 7{,}9375.10^{-3}\ m;$

$mb = 16{,}4.10^{-3}\ kg;$

$g = 9{,}81\ m/s^2.$

Para este experimento foi utilizada a técnica de controle fuzzy Takagi-Sugeno-Kang (ver Apêndice A).

No projeto de um controlador fuzzy devem ser determinadas as variáveis que podem ser medidas (entradas) ou controladas (saídas). Uma vez determinado um conjunto satisfatório de variáveis, torna-se necessário o conhecimento de intervalos, definindo-se assim as funções de pertinência para os valores que poderão ser assumidos por estas.

As variáveis de entrada e saída que são relevantes para o sistema proposto são:

- Deslocamento linear da bola (entrada 1);
- Velocidade linear da bola (entrada 2);
- Deslocamento angular da barra (entrada 3);
- Velocidade angular da barra (entrada 4);
- Força aplicada para manter a bola na barra (saída);

Inicialmente, as funções de pertinência das variáveis de entrada 63, 64, 65 e 66 estão configuradas respectivamente como x_1, x_2, x_3 e x_4, podendo ser observadas nas figuras abaixo:

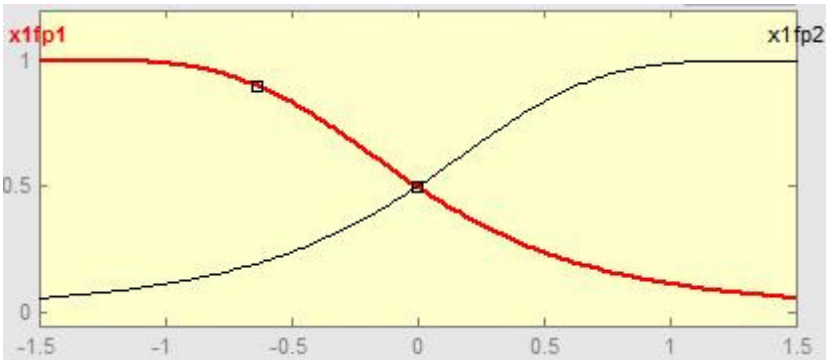

Figura 63 Função de Pertinência para o Deslocamento Linear da Bola

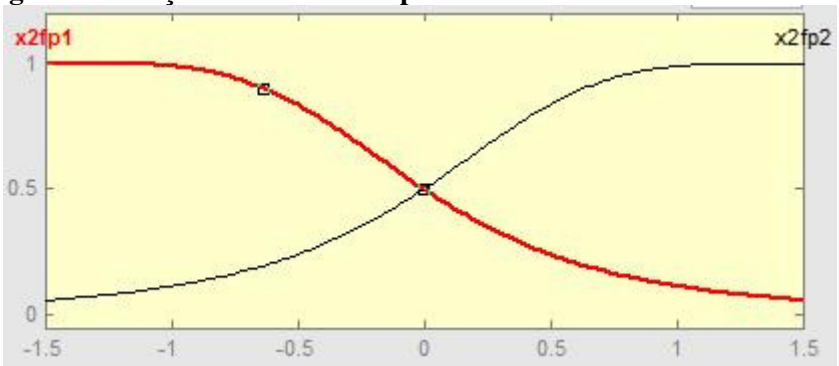

Figura 64 Função de Pertinência para a Velocidade Linear da Bola

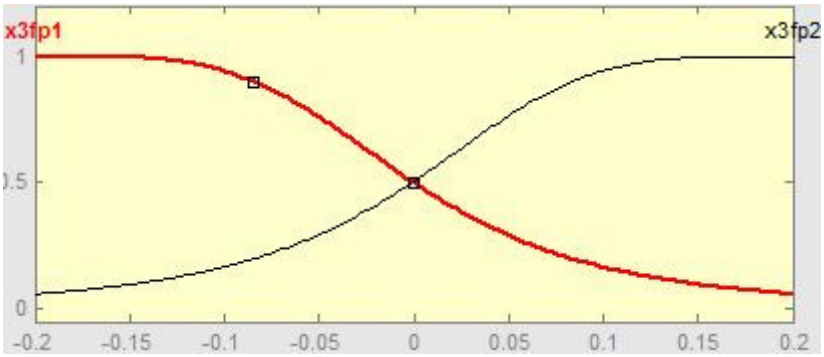

Figura 65 Função de Pertinência para o Deslocamento Angular da Barra

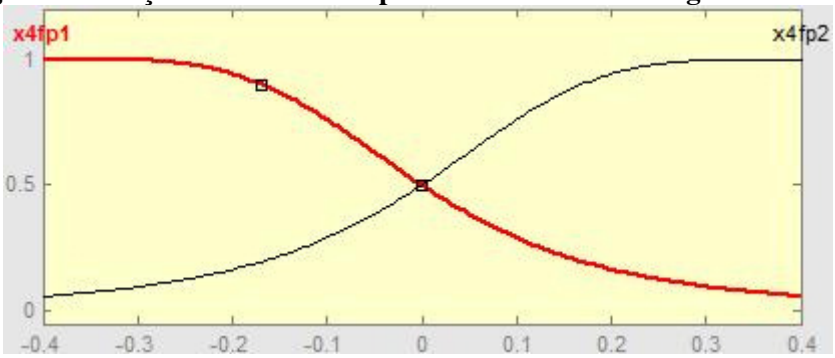

Figura 66 Função de Pertinência para a Velocidade Angular da Barra

Desta forma se propõe que o usuário observe o comportamento do sistema, alterando a referência de entrada e/ou as funções de pertinência das variáveis de entrada (x_1, x_2, x_3 e x_4) e observe a resposta apresentada pelo sistema Barra-Bola.

2 SISTEMA MASSA, MOLA E AMORTECEDOR

Este tipo de sistema é constituído de uma massa *m*, sobre a qual três forças podem ser aplicadas: a força *F*, provocando o deslocamento da massa e tendo como reações as forças da mola (*k*) e do amortecedor (*b*). A figura 67 representa o sistema amortecido de segunda ordem.

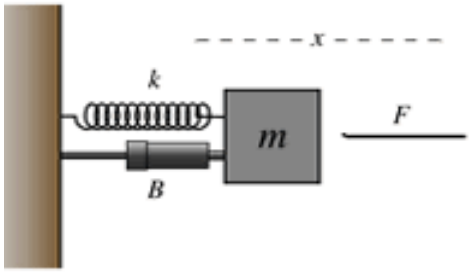

Figura 67 Modelo Massa, Mola e Amortecedor

A lei fundamental que governa sistemas mecânicos é a lei de Newton, sendo que para o tipo de sistema apresentado a lei estabelece que:

$$\sum F = ma \tag{D.3}$$

Ou seja, a resultante das forças é igual ao produto da massa pela aceleração. Assim, aplicando a lei de Newton no sistema apresentado na figura 67, obtêm-se:

$$m\frac{d^2y}{dt^2} = -b\frac{dy}{dt} - ky + x \tag{D.4}$$

Ou:

$$m\frac{d^2y}{dt^2} + b\frac{dy}{dt} + ky = x \tag{D.5}$$

Se considerarmos as condições iniciais nulas, a transformada de Laplace da equação (D.5) pode ser escrita como:

$$\left(ms^2 + bs + k\right)Y(s) = X(s) \tag{D.6}$$

E calculando a relação entre *Y(s)* e *X(s)* a função de transferência do sistema torna-se:

$$\frac{Y(s)}{X(s)} = \frac{1}{ms^2 + bs + k} \tag{D.7}$$

Onde:

m = 1 kg;

b = 10 N.s/m;

k = 20 N/m;

F(s) = 1.

Logo,

$$X(s) = \frac{1}{s^2 + 10s + 20} \tag{D.8}$$

Utilizando a RTAI, faz-se necessário realizar a discretização do sistema. Assim, utilizando 0.001s como período de amostragem, obteve-se:

$$X(z) = \frac{0.001z + 0.001}{z^2 - 1.9425z + 0.9425} \tag{D.9}$$

A figura 68 apresenta o diagrama de blocos de um controlador PID discreto.

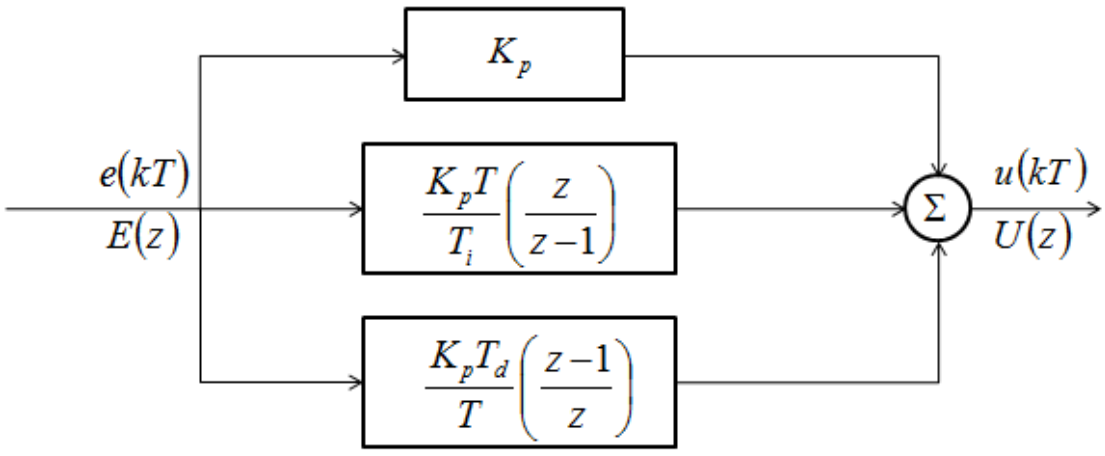

Figura 68 Diagrama de Blocos de um Controlador PID Discreto

Os valores apropriados para *Kp, Ti* e *Td* podem ser determinados pelo método da resposta transiente, proposto por Ziegler e Nichols, conforme descrito na seção de sistemas de controle.

Neste experimento se propõe que usuários alterem a referência, assim como os ganhos Proporcional (*Kp*), Integral (*Kd*) e Derivativo (*Ki*), observando a resposta do sistema.

3 CONTROLE DE POSIÇÃO DE UM MOTOR

Este experimento simula o controle de posição de um motor DC (MICHIGAN, 2009). Este motor provê diretamente movimento rotacional e, acoplado a rodas ou tambores e cabos, pode prover movimento transacional. O circuito elétrico da armadura e o diagrama do rotor são apresentados na figura 69:

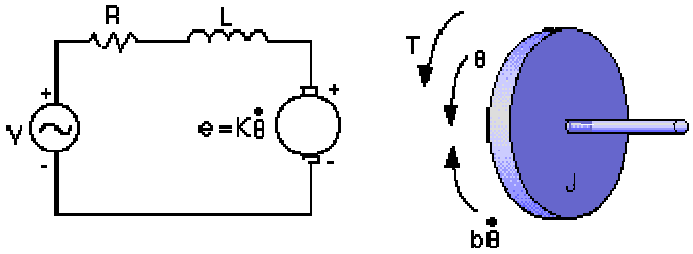

Figura 69 Circuito Elétrico da Armadura e Diagrama do Rotor de um motor DC
Fonte: MICHIGAN, 2009.

As equações dinâmicas na forma de função de transferência são representadas por:

$$\frac{\theta}{V} = \frac{K}{s\left((Js+b)(Ls+R)+K^2\right)} \qquad \text{(D.10)}$$

Para este exemplo, os valores escolhidos para as variáveis foram:

J (momento de inércia do motor) = 0.0000032284 kg.m²/s²;
b (razão de amortecimento do sistema mecânico) = 0.0000035077 Nms;
K(força constante eletromotriz) = 0.0274 Nm/Amp;
R (resistência elétrica) = 4 ohm;
L (indutância elétrica) = 0.00000275 H;

V (entrada) = fonte de voltagem;

θ (saída) = posição da haste;

Utilizando a RTAI faz-se necessário realizar a discretização do sistema, por isso utilizou-se 0.001s como período de amostragem, obtendo-se:

$$X(z) = \frac{0{,}001z + 0{,}001z}{z^2 - 1{,}9425z + 0{,}9425} \tag{D.11}$$

Neste experimento se propõe que usuários alterem a referência, assim como os ganhos Proporcional (*Kp*), Integral (*Kd*) e Derivativo (*Ki*), observando a resposta do sistema.