Editor de provas do Sistema Heráclito: Ferramenta de apoio ao Ensino da Dedução Natural na Lógica Proposicional

Fabiane Flores Penteado Galafassi[1], Rafael Koch Peres[2], Alan Velasques Santos[2], Rosa Maria Vicari[2], João Carlos Gluz[3]

[1] PPGIE – UFRGS, CEP: 90040-060, Porto Alegre, RS – Brasil
[2] PPGC – UFRGS, CEP: 91501-970, Porto Alegre, RS – Brasil
[3] PIPCA – UNISINOS – Caixa Postal 275 – 93.022-000 – São Leopoldo – RS – Brasil

{fabiane.penteado, rafaelkperes, alanvelasques}@gmail.com,
rosa@inf.ufrgs.br, jcgluz@unisinos.br

Abstract. *This article aims to present the evidence Editor Heraclito system. The Heraclito is an Intelligent Tutor System (ITS) facing the propositional logic teaching for undergraduates in Computer. His evidence editor's main functionality help to develop evidence of formal arguments by the rules of the Natural Deduction. Your current interface version was developed using the technique of Test Driven Development (TDD) and access is via Web Browser in order to make adaptive tool and may be accessed through computers, tablets, mobile phones and even SmartTVs.*

Resumo. *O presente artigo tem como objetivo apresentar o editor de provas do sistema Heráclito. O Heráclito é um Sistema Tutor Inteligente (STI) voltado para o ensino de lógica proposicional para alunos de graduação em Computação. Seu editor de provas tem como principal funcionalidade auxiliar na elaboração de provas de argumentos formais por meio das regras da Dedução Natural. Sua atual versão de interface foi desenvolvida utilizando a técnica de Test Driven Development (TDD) e seu acesso é feito via Web Browser, de forma a tornar a ferramenta adaptativa, podendo ser acessada através de computadores, tablets, celulares e até mesmo SmartTVs.*

1. Introdução

De acordo com o currículo oficial do Ministério da Educação (MEC) do Brasil, a disciplina de Lógica[1], abordada no ensino superior, é uma disciplina básica e obrigatória para todos os cursos que abrangem computação e informática. Fundamental para a formação dos alunos que cursam essas graduações, a disciplina de Lógica possibilita o desenvolvimento das habilidades de análise lógica, formalização e resolução de problemas. Habilidades, que por sua vez, são necessárias para a compreensão dos diversos conteúdos e atividades encontradas nos componentes curriculares da Computação e Informática.

Um levantamento estatístico[2] realizado nos últimos oito anos apontou para índices muito altos de reprovação e desistência nesta disciplina. Esses índices, em

[1] Parecer CNE/CES nº 136/2012, aprovado em 8 de março de 2012 - Diretrizes Curriculares Nacionais para os cursos de graduação em Computação. Disponível em:<http://portal.mec.gov.br/index.php?option=com_docman&task=doc_download&gid=11205&Itemid

[2] Os dados compilados para este estudo e que justificaram o desenvolvimento de uma ferramenta para o ensino de Lógica se encontram em: http://biblioteca.asav.org.br/vinculos/000003/00000335.pdf, página 15, item 2.1 Justificativa.

DOI: 10.5753/cbie.wcbie.2015.455

particular, tendem a ocorrer no início da disciplina, principalmente quando os conteúdos de Dedução Natural começam a ser abordados no contexto da Lógica Proposicional. Na prática, as dificuldades começam quando conceitos como fórmula, regra de dedução e prova formal começam a ser apresentados. De forma a contribuir na melhoria dos índices apresentados, um método dialético de ensino foi utilizado, embasado em uma pedagogia sócia histórica, e um modelo de mediação por computador, modelado em um sistema multiagente foi projetado. Esse modelo de mediação foi o produto final apresentado em dissertação de mestrado e vem sendo adaptado, melhorado e desenvolvido nos dois últimos anos como parte de tese de doutorado, utilizando novas tecnologias do campo da IA, pelo Grupo de pesquisa GIA[3]. Esse sistema foi denominado de sistema Heráclito e possui até o momento um editor de provas formais.

O sistema Heráclito é um objeto de aprendizagem (OA) que tem como objetivo apoiar o ensino de Lógica. Seu editor de provas tem como principal funcionalidade auxiliar na elaboração de provas de argumentos formais por meio das regras da Dedução Natural na lógica Proposicional. Tem seu foco associado ao uso de tecnologias de agentes com características pedagógicas que fazem o uso de diversas estratégias de ensino-aprendizagem. O intuito é o de auxiliar o aluno em seu processo de raciocínio na resolução de um exercício.

Inicialmente projetado para as versões Desktop/Laptop e Móvel (*Tablets* e *Smartphones*), sua versão atual foi unificada desenvolvida utilizando a técnica de Test Driven Development (TDD)[4] e Java. Seu acesso é feito via *Web Browser* (Navegadores *Web*) com características responsivas. Sua licença de uso é Software gratuito (Freeware).

Desta forma, neste contexto, o artigo está organizado como segue: no capítulo 2 aborda-se o desenvolvimento do editor de provas do Sistema Heráclito e sua relação com a infraestrutura OBAAMILOS[5]. No capítulo 3 é feita a apresentação do editor de provas do Heráclito com apresentação dos testes realizados no semestre de 2015/1. No capítulo 4 são feitas algumas considerações finais com relação à ferramenta e também são apresentados trabalhos futuros que já estão em fase de desenvolvimento/aperfeiçoamento, tanto pedagógico quanto tecnológico.

2. Desenvolvimento do Sistema Heráclito, seu Editor de Provas e a Infraestrutura OBAAMILOS

O sistema Heráclito é integrante do projeto OBAAMILOS como parte do Sistema de Apoio Pedagógico. Foi construído através do uso da tecnologia de agentes pedagógicos (Galafassi, 2013), responsáveis pela interação do aluno com o sistema (editor de provas). Possui ainda acesso a um tutor inteligente que auxilia no desenvolvimento das provas, indicando caminhos corretos e incorretos e não recomendáveis, dando dicas de passos e regras, indicando exemplos de provas já solucionadas e materiais complementares durante o andamento da prova. Estas características de arquitetura diferenciam o Heráclito de um provador de teoremas e o transformam em um STI.

[3] Mais informações sobre o grupo de pesquisa em: http://gia.inf.ufrgs.br/portal/index.php?q=pt-br.

[4] Test Driven Development, em português Desenvolvimento guiado por testes, refere-se a um estilo de programação no qual três atividades estão estreitamente entrelaçadas: codificação, testes (na forma de testes de unidade escritas) e design (na forma de refatoração). Mais informações em português podem ser visualizadas em: http://tdd.caelum.com.br/.

[5] Mais informações em: http://www.portalobaa.org/padrao-obaa/relatorios-tecnicos/copy_of_1o-relatorio-parcial-obaa-milos-comunidade-finep/AnexoA-EspecArqMILOSV10.pdf/view.

2.1 Infraestrutura MILOS

A infraestrutura MILOS é uma infraestrutura de agentes que suporta a autoria, gerência, busca e disponibilização de objetos de aprendizagem compatíveis com o OBAA (Objetos de aprendizagem Baseados em Agentes Artificiais), e tem por objetivo mais geral do projeto Infraestrutura OBAA-MILOS criar as bases tecnológicas que permitam efetivar a adoção da proposta de metadados de objetos de aprendizagem OBAA (Gluz, 2010).

A MILOS foi projetada em três grandes níveis de abstração: Nível das Ontologias, Nível de Agentes, Nível das Facilidades de Interface. A camada de nível de agentes implementa o suporte as atividades de suporte ao ciclo de vida de um OA, incluindo suporte para autoria, adaptação, gerenciamento, publicação, localização e uso de OAs compatíveis com o OBAA. Esse suporte foi distribuído em quatro grandes sistemas multiagente: (1) Sistema de Busca Federada, que suporta as atividades de localização dos OAs, (2) Sistema de Apoio Pedagógico, que fornece apoio ao uso dos OAs em ambientes de ensino, (3) Sistema de Autoria, responsável pelo apoio as atividades de autoria de OAs, incluindo suporte a adaptação multiplataforma, e (4) Sistema de Gerência, responsável pelas atividades de armazenamento, gerenciamento, publicação/distribuição multiplataforma de OAs. Na prática, o Sistema Heráclito é um protótipo inicial do Sistema de Apoio Pedagógico da infraestrutura MILOS (Vicari, 2010).

2.2 Integração do sistema Heráclito e seu editor de provas com a Interface *Web*

O sistema Heráclito é composto por uma interface e um editor de provas, ambos desenvolvidos em Java, e por seu conjunto de agentes pedagógicos: perfil do aluno também desenvolvido em Java, agente mediador desenvolvido em AgentSpeak(L) e agente especialista desenvolvido em Prolog (Galafassi, 2012).

O sistema Heráclito é acessado via *Web*, e sua integração se deu diretamente por meio dos agentes pedagógicos, abstraindo as camadas de interface anteriores e do editor de provas. Essa integração da interface com os componentes do sistema Heráclito foi divida em três partes: interface, lógica (editor de provas) e a integração, onde as duas primeiras são independentes e a última depende do término das duas primeiras.

Para a parte lógica, manteve-se a linguagem Java, uma vez que será desenvolvido para um serviço *Web*, sendo uma aplicação cliente-servidor e pouco sensível à linguagem escolhida, necessitando apenas de uma linguagem com suporte a *Web* e orientada a objetos para fins de organização.

Como base para o desenvolvimento, adotou-se a técnica *Test Driven Development* (TDD), onde são definidos testes para cada unidade e esta é desenvolvida a fim de satisfazer estes testes, garantindo sua funcionalidade e evitando erros em sua futura integração no sistema. Para o auxílio nestes testes, adotou-se o *framework* JUnit. Neste, para se testar a aplicação de uma regra, primeiro se verifica se o método de aplicação aceita apenas o número de linhas correto para esta regra; então se testa uma série de entradas válidas e inválidas, ignorando-se a saída, apenas avaliando a aceitação e rejeição das entradas; por último são avaliadas as saídas de acordo com as entradas, verificando se a aplicação está correta e se a linha resultante foi definida como resultado da regra correta (esta aparecerá na prova) (Astels, 2003). Para que a interface se ligue à parte lógica, utilizou-se *Java Server Pages* (JSP), sendo este semelhante a outras

linguagens *back-end* de auxílio ao desenvolvimento de aplicações para *Web* e utiliza para isto a linguagem Java que poderá comunicar diretamente com os agentes pedagógicos já desenvolvidos do Heráclito.

A escolha da arquitetura cliente-servidor para o desenvolvimento da interface do Heráclito se deu em virtude da segurança e desempenho que essa metodologia, associada a linguagens de programação preparadas, pode oferecer para o sistema, tendo em vista que essa interface foi desenvolvida com o objetivo de serem portável para diferentes dispositivos com diferentes capacidades, tamanhos de telas e desempenhos diferentes.

Na figura 1 é possível ver a existência de dois pontos críticos em desempenho e segurança que são: a linguagem Java onde está a lógica do programa, e os agentes em si que vão entrar em contato com o usuário. Com a utilização da arquitetura cliente-servidor, é possível enviar para o usuário apenas a resposta da requisição em HTML, CSS e Javascript deixando toda a parte lógica do provador e toda a parte de agentes sendo processado inteiramente dentro do servidor, o que torna o sistema extremamente leve e portável para qualquer dispositivo que possuir um navegador *Web*.

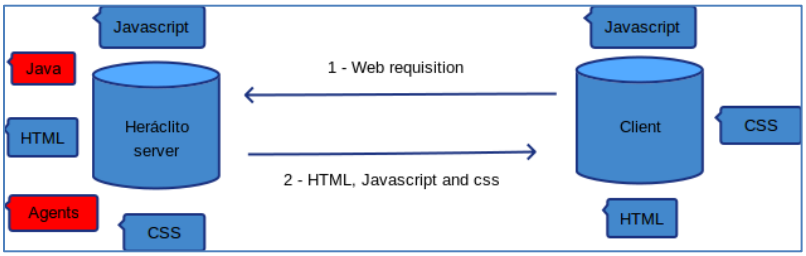

Figura 1. Representação da arquitetura cliente servidor do novo Heráclito.
Fonte: Elaborado pela autora e demais integrantes do projeto OBAAMILOS

Visando o aumento da segurança, robustez, compatibilidade com diversos sistemas e tamanhos de tela, foi utilizada para desenvolvimento o *framework Bootstrap*[6]. Este *framework* fornece uma gama de elementos prontos em HTML 5, CSS 3 e *JavaScript* que auxiliam na adaptação do site a diversos tamanhos de tela e diversos sistemas tornando-os responsivos, o que faz com que o objeto de aprendizagem possa, com apenas uma versão, ser distribuído para diversas plataformas diferentes alcançando um número maior de usuários.

3. Apresentação do editor de provas do sistema Heráclito

O Heráclito consiste em um editor de provas que aborda o conteúdo de Dedução Natural na Lógica proposicional (DNLP) e que utiliza como apoio pedagógico um Tutor no acompanhamento e desenvolvimento da realização das provas.

Atualmente o sistema tutor encontra-se em fase de desenvolvimento e manutenção em sua plataforma de agentes. Dentre as mudanças de linguagens de programação, outros serviços e recursos estão sendo adicionados e melhorados, incluindo seu manual e guia de utilização.

3.1 Interface inicial

[6] Bootstrap. Framework *Web* gratuito da empresa Twitter. Disponível em: <http://getbootstrap.com/>.

A página inicial apresenta uma tela de boas vindas, oferecendo uma breve ideia do objetivo e estado atual do editor de provas do Heráclito. O *login* para acesso se encontra no canto superior direito para acessar o sistema e poder utilizar o editor com suporte ao tutor pedagógico, a descrição do sistema e também a tela para cadastro de novos usuários (opção obrigatória para utilizar os recursos de tutoria). Caso os recursos de tutoria não sejam desejáveis é possível utilizar apenas o editor de provas. Um link para acesso ao editor como visitante está disponível.

Ao acessar o editor de provas como visitante, um menu de usuário será oferecido com quatro opções: *Nova Prova, Sobre, Manual* e *Sair*.

A Figura 2 mostra as quatro opções do editor de provas:

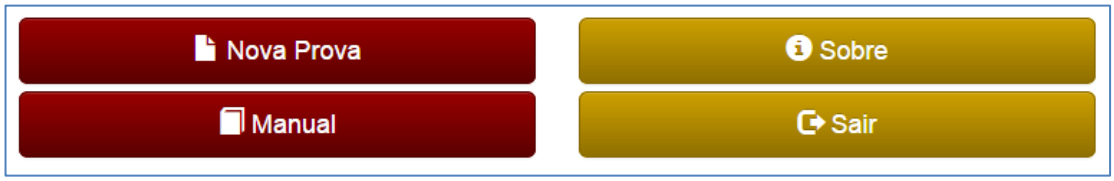

Figura 2. Tela de opções do editor de provas acessado via *Desktop/Laptop*.
Fonte: Fonte: http://heraclito.inf.ufrgs.br/heraclito/

Na opção *Sobre* será aberta uma janela (modal) com breves dados do projeto Heráclito. Na opção *Manual* é oferecido um link externo para o manual do sistema (em pdf). A opção *Sair* irá (des)autenticar o usuário atual do sistema (editor de provas), limpando a prova em andamento. A opção *Nova Prova* será exibida uma interface para a escolha da proposição a ser provada.

A edição das provas se dá por meio da opção *Nova Prova*, é por meio desta interface que o editor de provas do Heráclito é executado. Na interface de escolha da proposição a ser provada, é possível iniciar uma prova a partir de uma proposição personalizada, digitando-a no campo *Digite o cabeçalho para a nova prova* e clicando em *Começar*. Ao iniciar uma prova personalizada, cuidado ao digitá-la, pois alguns erros de digitação poderão acontecer e serem considerados como uma proposição correta pelo sistema.

As figuras 3 e 4 apresentam a interface do editor de provas fazendo o acesso utilizando um *Tablet* e um *Ipad*.

Também é possível optar em utilizar provas já pré-selecionadas pelo conjunto de provas ordenadas por níveis: *Provas Básicas*, *Provas Intermediárias* e *Provas Avançadas*. Cada conjunto pode ser acessado expandindo sua aba de mesmo nome. Dentro de cada aba é oferecida uma *lista de seleção*. Após escolhida a proposição desejada da lista de seleção, basta apenas clicar em *Começar*.

A opção *Nova prova* poderá aparecer como *Continuar Prova* caso uma prova esteja em andamento.

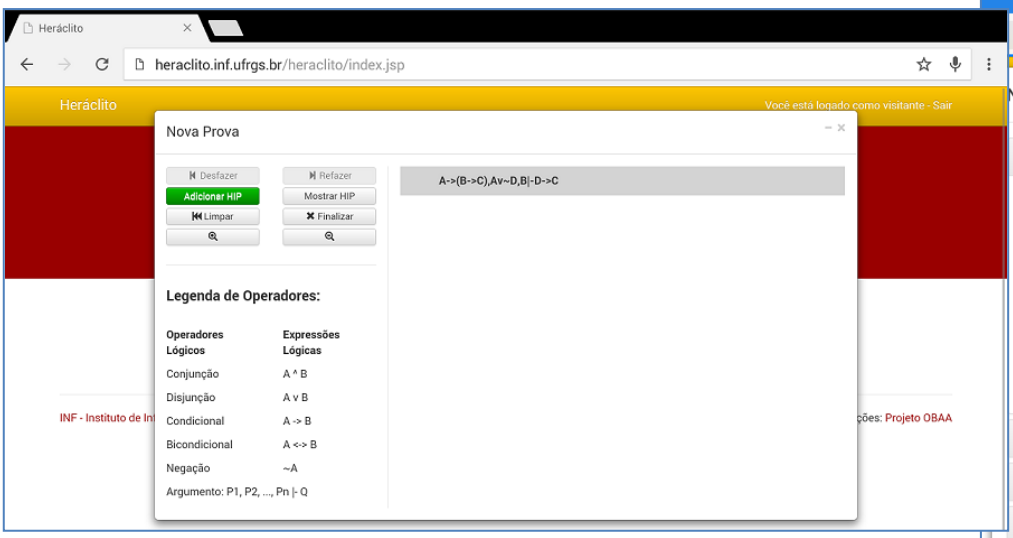

Figura 3. Interface do editor de provas acessado via *Tablet*
Fonte: Fonte: http://heraclito.inf.ufrgs.br/heraclito/

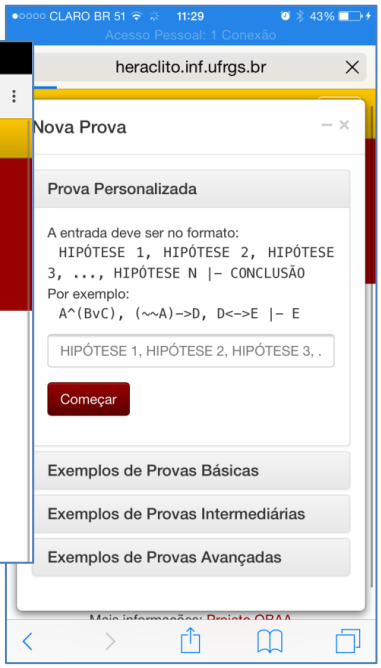

Figura 4. Interface do editor de provas acessado via *Iphone*
Fonte: Fonte: http://heraclito.inf.ufrgs.br/heraclito/

A resolução de uma prova neste sistema se dá em dois passos principais, necessariamente nessa ordem: adição de hipóteses, caso estas existam, e aplicação de regras. Uma prova é considerada concluída ao, após se aplicar uma série de regras, chegar-se à expressão procedente ao materlo lógico (⊢), e.g.: A, B ⊢ A^B, onde A^B é a expressão procedente e A e B são as hipóteses.

As figuras 5 e 6 apresentam o editor de provas sendo executado e seu acesso utilizando um *Tablet* e um *Ipad*.

O campo de texto Linha e o campo de seleção Esquerda/Direita não são necessários para todas as ações, apenas para as quais tiverem estes campos especificados neste manual (caso o Linha seja necessário e não esteja preenchido, o sistema informará).

A demonstração das provas são elaboradas passo-a-passo com base na aplicação das regras de inferência (básica e derivada).

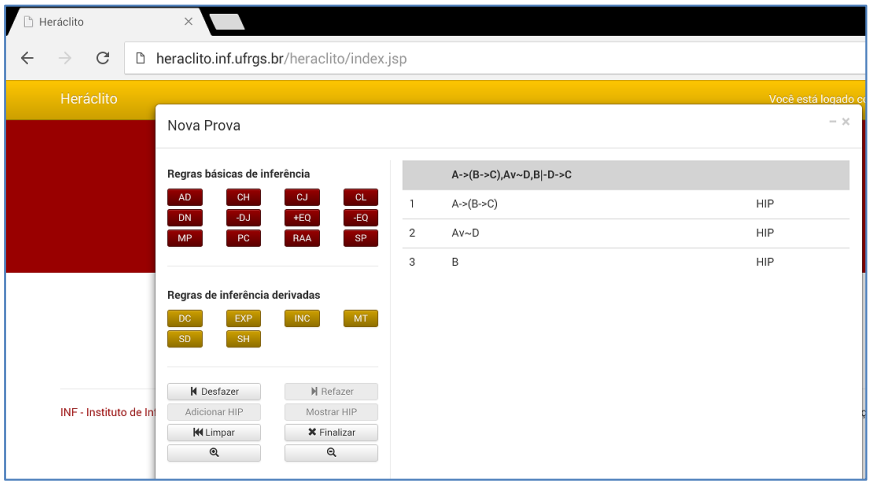

Figura 5. Tela de edição de provas acessado via *Tablet*
Fonte: Fonte: http://heraclito.inf.ufrgs.br/heraclito/

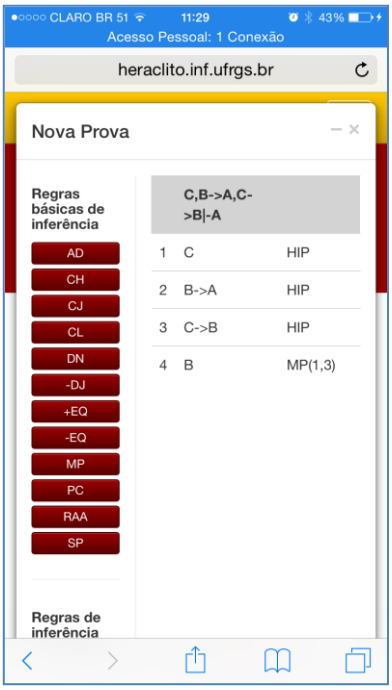

Figura 6. Tela de edição de provas acessado via *Ipad*
Fonte: Fonte: http://heraclito.inf.ufrgs.br/heraclito/

São oito regras básicas de inferência não hipotética: MP (modus ponens), DN (dupla negação ou eliminação Da negação), CJ (introdução da conjunção), SP (separação ou eliminação da conjunção), AD (adição ou introdução da disjunção), DJ (eliminação da disjunção), +EQ (introdução da equivalência ou do bicondicional), -EQ (eliminação da equivalência ou do bicondicional). E duas regras hipotéticas: RAA (redução ao absurdo) e PC (prova condicional), além das regras derivadas: MT (modus tolens), SH (silogismo hipotético), SD (silogismo disjuntivo), DC (dilema construtivo), INC (inconsistência) e EXP (exportação) (Galafassi, 2012).

Outras opções são demonstradas no manual de utilização do editor de provas do sistema Heráclito e podem ser acessadas no endereço: http://heraclito.inf.ufrgs.br/heraclito/resources/manual.pdf.

4. Considerações finais e trabalhos futuros

O editor de provas quando utilizado em modo local, como é o atual *status* apresentado pela ferramenta, ainda é capaz de tratar algumas situações que ocorrem durante a resolução de um problema de dedução como, por exemplo, a verificação da inserção de hipóteses, onde o editor verifica se a hipótese é correta ou não. A verificação de aplicação das regras de dedução, onde o editor simplesmente não permite que o aluno insira um novo passo através da aplicação incorreta de uma regra de dedução. E em modo local o botão Ajuda provê acesso ao manual do editor e a exemplos de provas previamente resolvidas. O exemplo é escolhido aleatoriamente porque o Editor não possui suporte do serviço de tutoria para saber qual exemplo de uso de regra de dedução sugerir para o próximo passo.

Em testes realizados anteriormente[7], nas versões para *desktop* (*applet* Java) e móvel (*app*), obteve-se um resultado de Efetividade Pedagógica de Mediação com média de 92% de aprovação. Quanto a Qualidade do conteúdo apropriada para o ensino de dedução natural na Lógica Proposicional, obteve-se um total de 88% de concordância em geral sobre a boa qualidade (resultado importante porque demonstra a aplicabilidade do OA para fins pedagógicos). Quanto a Qualidade do Layout Visual, pode-se verificar que a grande maioria dos alunos acredita que a ferramenta se mostra intuitiva, e que de algum modo auxilia no processo de aprendizagem, conforme 80% dos alunos. E quanto ao quesito Usabilidade, nesta etapa foi possível avaliar as questões de âmbito gerais, onde se obteve uma média de 85% e em questões específicas sobre a usabilidade, obtendo uma média de 84% de concordância. Em ambos os casos o sistema Heráclito obteve boa aceitação por parte dos alunos. Em sua nova versão para *Web* onde sua interface foi integrada ao editor, associando conceitos de adaptabilidade e portabilidade, espera-se que os níveis de aceitação da ferramenta se mantenham ou até mesmo superem os anteriores. De qualquer forma, o processo de mediação no Sistema Heráclito é um método de ensino implementado computacionalmente e sofre das mesmas dificuldades de avaliação de sua eficácia.

Como trabalhos futuros pretende-se terminar a implementação da nova versão, fazendo a integração da interface html/css/js com a lógica (Java) já implementadas e os agentes em seus novos formatos com as mesmas funcionalidades e recursos anteriormente. Como novos recursos, adicionar-se-á a opção de salvar a prova atual e continuá-la posteriormente; a internacionalização do Heráclito, possibilitando a exibição da página em pelo menos mais dois idiomas; e a opção de acessar recursos off-line do Heráclito, necessitando de conexão à internet apenas para o acesso à página inicial.

Os testes de usabilidade com a nova interface estão previstos para o inicio do semestre de 2015/2 com turmas de graduação nas IES: Universidade Federal do Rio Grande do Sul e Universidade do Vale do Rio dos Sinos. Para 2016/1 pretende-se integrar o Heráclito com o sistema SAAPIENS[8] e posteriormente desenvolver um portal especifico para o ensino de Lógica contemplando outros conteúdos além da DNLP.

5. Referencias

Astels, David. Test-Driven Development: A Practical Guide: A Practical Guide. 2003.

Galafassi, F.F.P. (2012) "Agente Pedagógico para Mediação do Processo de Ensino-Aprendizagem da Dedução Natural na Lógica Proposicional". Dissertação de Mestrado. UNISINOS.

Galafassi, Fabiane F. P., Gluz, J. C., Gomes, Lucas, Mossmann, Marcel. "Sistema Heráclito: Objeto de Aprendizagem para o Ensino da Dedução Natural na Lógica Proposicional". Anais do CBIE, 2013.

Gluz, J. C.; Vicari, R. M. MILOS: Infraestrutura de Agentes para Suporte a Objetos de Aprendizagem OBAA. Anais do XXI SBIE, 2010.

Gluz, J.C.; PY, M. (2010) Lógica para Computação. Coleção EAD. Editora Unisinos.

Vicari, R.; Gluz, J.; Passerino, L.; et al. (2010): The OBAA Proposal for Learning Objects Supported by Agents. Procs. Of MASEIE Workshop – AAMAS 2010, Toronto, Canadá.

[7] Dados podem ser visualizados em: http://biblioteca.asav.org.br/vinculos/000003/00000335.pdf, página 95, item 7.
[8] Mais informações sobre o sistema SAAPIENS em: http://www.br-ie.org/pub/index.php/sbie/article/view/3041/2552