View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by El Servicio de Difusión de la Creación Intelectual

DESENVOLVIMENTO DE UM COMPONENTE PARA PUBLICAÇÃO E REVISÃO DE ARTIGOS (*PAPER COMPONENT*)

José Roberto Vasconcelos[1] , Ivan Luiz Marques Ricarte[2]
Rafael Alessandro Gatto[1], Gislaine Camila Lapasini Leal[1]
Márcia Toshie Tanimoto[1]

[1] *Departamento de Informática, Universidade Estadual de Maringá*
Maringá – PR – Brazil

[2] *Departamento de Engenharia de Computação e Automação, Universidade Estadual de Campinas*
Campinas – SP – Brazil

E-mail: [1] {jrvasco, ragatto, camila, marciatan}@din.uem.br, [2] ricarte@dca.fee.unicamp.br

IV – Workshop de Tecnología Informática Aplicada en Educación

Abstract –The environment development of Education in the distance has been stimulated for the diffusion of the long-distance education in almost the whole world, the increasing technological evolution, and the expansion of the InterNet. In this way, the article presents a component for publication and article revision, the "Paper Component", describing its functioning; carrying through a Guided modeling the Objects; showing a vision through components; the possible forms of adaptation of exactly in EAD environments; e some interfaces of the archetype bred.

Resumo – O desenvolvimento de ambientes de Educação a Distância tem sido impulsionado pela difusão da educação à distância em quase todo o mundo, a crescente evolução tecnológica, e a expansão da internet. Desse modo, o artigo apresenta um componente para publicação e revisão de artigos, o "Paper Component", descrevendo o seu funcionamento; realizando uma modelagem Orientada a Objetos; mostrando uma visão através de componentes; as possíveis formas de adaptação do mesmo em ambientes de EAD; e algumas interfaces do protótipo criado.

Palavras Chaves – Componente, educação a distância, reuso, adaptação.

1. INTRODUÇÃO

A Educação a Distância (EAD) vem crescendo no âmbito educacional, e é uma forma de ensino em que professores e alunos se encontram separados geograficamente e/ou temporalmente. A EAD está sendo difundida em quase todo o mundo, desde as nações industrializadas até os países em desenvolvimento.

Em se tratando de ensino a distância, somente os recursos textuais, áudio e vídeo, já não são o suficiente, com isto torna-se necessário o estabelecimento de uma comunicação de via dupla, o que requer a utilização de meios como a Internet, que possibilite a comunicação entre ambos de forma eficaz [1].

A EAD é uma forma de *e-learning* por basear-se em encontros presenciais e na grande maioria das vezes virtuais onde os alunos dos cursos vivenciam novas experiências, são autores de sua aprendizagem, pois eles são os próprios construtores de seu desenvolvimento cognitivo, é o aprender a aprender [2].

Com a rápida e crescente expansão da internet, diversos ambientes de autoria para a criação e aplicação de cursos à distância foram desenvolvidos, tais como: WebCT, AulaNet, LearningSpace,

TelEduc e Moodle. Esses ambientes oferecem ferramentas que auxiliam a interação entre professores e alunos, fornecendo espaços para explorar, expressar, comunicar idéias, passar informações e consequentemente produzir conhecimento [1].
O presente artigo propõe-se a apresentar o componente *Paper Component* desenvolvido descrevendo o seu funcionamento; retratando os aspectos estáticos e dinâmicos através de alguns diagramas da *Unified Modeling Language* (UML); realizando uma abordagem segundo a Arquitetura Proposta em [5]; mostrando interfaces do protótipo construído; e mostrando as possíveis formas de adaptação do mesmo em diversos ambientes de EAD.
Nesse contexto esta pesquisa objetiva desenvolver um componente, unidade de software independente, para auxiliar o desenvolvimento de artigos e o compartilhamento desses em ambientes de EAD.

2. DESCRIÇÃO DO *PAPER COMPONENT*

Uma das maiores vantagens na utilização de componentes de software é a criação de um mercado, que possibilita com que os desenvolvedores de software possam comprar os componentes e assim criar de maneira mais rápida os softwares [7].
Um ponto essencial na abordagem de desenvolvimento baseado em componentes é a forma de realizar a conexão de componentes. Levando-se em conta um ambiente de desenvolvimento em que todos os componentes tenham sido implementados em uma mesma linguagem de programação e possuam a mesma localização física, mecanismos da própria linguagem podem ser usados para que um componente mantenha referência de outro, e chamadas de procedimento dão acesso aos serviços de componentes [8].
A conexão de componentes ocorre desde que suas interfaces sejam compatíveis, isto significa que independentemente de homogeneidade de linguagem em que estão implementados, de localização física e de plataforma de execução, os serviços requeridos por um, estejam disponíveis no outro. A descrição de componentes deve permitir verificar esta compatibilidade, senão é impossível verificar se dois componentes podem ser conectado[8].
O *Paper Component* visa facilitar a tarefa de publicação, revisão e leitura de artigos num ambiente de EAD. Ele encapsula dentro de si seu projeto e sua implementação, e oferece pontos de interconexão, as chamadas Interfaces Fornecidas ou Interfaces Requeridas, para comunicação com os ambientes de EAD [3].
Como Interface Requida, aquela que define as operações que o componente requisitará do ambiente de EAD para seu funcionamento, tem-se os dados (id, nome, login e senha) do aluno e do administrador. Como Interface Fornecida, aquela que contém as operações que o componente fornece para os demais componentes, os artigos com suas respectivas correções/observações.
O funcionamento do componente ocorre quando um aluno, na função de escritor, disponibiliza o artigo para correção. Após disponibilizar o artigo, inicia-se a tarefa de distribuição, a qual é feita segundo os parâmetros (número de revisores, número de artigos por revisor, forma de distribuição) de configuração do componente.
Na etapa de distribuição, o aluno, enquanto o revisor, recebe o artigo e terá um determinado prazo para revisão e devolução do mesmo. Na fase de revisão, o revisor poderá adicionar comentários, sugestões e decidir em disponibilizar o artigo no ambiente ou devolver ao escritor para realizar eventuais correções. O artigo terá um número máximo de devoluções, sendo ultrapassado esse número o mesmo será excluído automaticamente.

3. MODELAGEM DO *PAPER COMPONENT* UTILIZANDO UML

O *Paper Component* foi modelado utilizando a UML, visto que a mesma é uma linguagem padronizada para documentar projetos de software. Ela surgiu como padrão para consolidar a notação utilizada em vários métodos (OMT, Booch, Jacobson, Fusion) existentes inicialmente.
Ela provê um vocabulário e um conjunto de regras para combinar os elementos da linguagem, focando os elementos conceituais e físicos que representam um sistema. A documentação UML permite retratar artefatos como requisições de negócios, modelo de arquitetura, código-fonte, modelo de análise, protótipo e outros documentos que servem de informação sobre o sistema [4].
A modelagem UML possibilita comunicar a estrutura e o comportamento do componente, visualizar e controlar a arquitetura, administrar riscos, facilitar o entendimento, expondo oportunidades para melhoras e reutilização [4].
Com o intuito de modelar o comportamento do sistema, fez-se a identificação dos casos de uso e atores envolvidos no componente de publicação de artigo. Os casos de uso representam a interação dos atores com o componente, como pode ser observado na Figura 1.

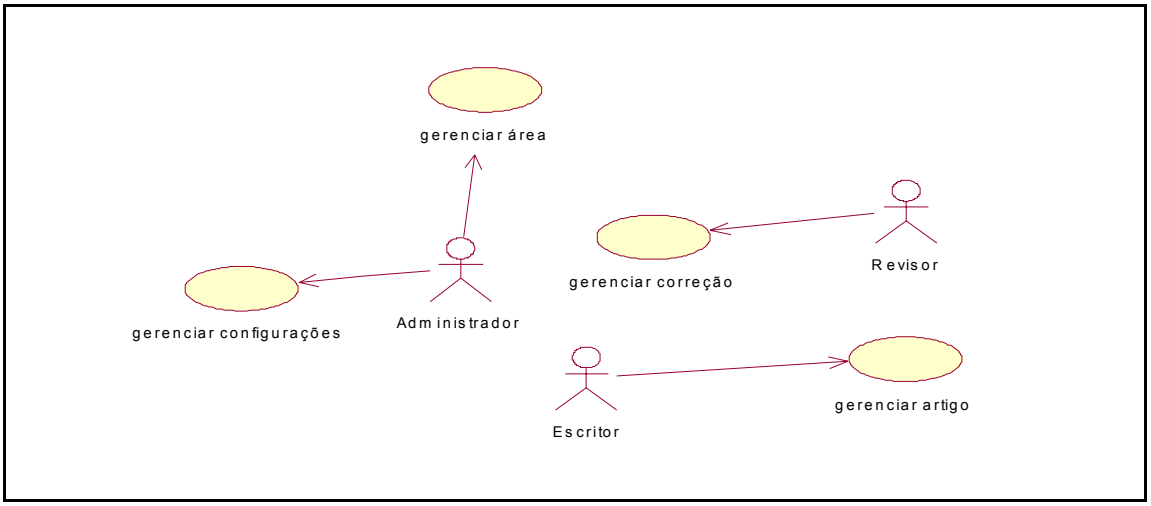

Figura 1: Diagrama de Caso de Uso geral do componente.

Na Figura 2, a partir da visão geral do componente, foram identificados os pacotes e os mecanismos de propósito geral que servem para agrupar os elementos da modelagem que estão semanticamente próximos e tendem a se modificar em conjunto.

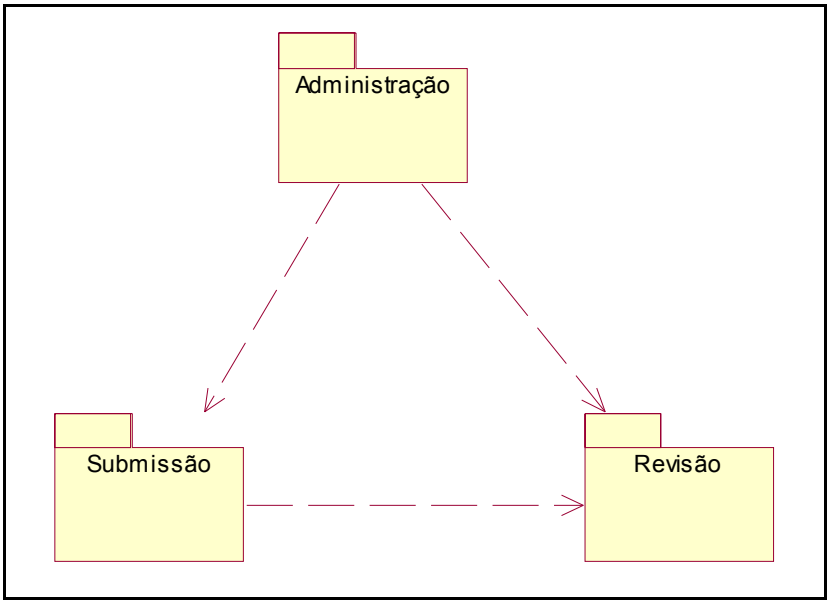

Figura 2: Visão de pacotes.

O pacote Administração engloba os casos de uso: gerenciar configurações e área (pesquisa/ensino). Neste pacote somente o ator administrador interage com o componente.
No pacote submissão tem-se o caso de uso gerenciar submissão o qual é realizado pelo ator escritor.
O pacote revisão contém o caso de uso gerenciar revisão, que é realizado pelo ator revisor.
A seguir estes pacotes serão mais detalhados.

3.1 PACOTE ADMINISTRAÇÃO

A Figura 3 descreve as interações que o ator, administrador, realiza no pacote de Administração. Neste cenário o administrador tem a possibilidade de alterar as configurações *default* do ambiente, realizar as operações de cadastro/alteração/exclusão das áreas de ensino/pesquisa.
O caso de uso alterar configurações permite ao ator definir o modo de operação do componente através dos seguintes parâmetros: separar artigos/usuários por área; número mínimo e máximo de revisores; prazo para revisão; número máximo de revisões; possibilidade do revisor selecionar o artigo; encaminhar mensagem solicitando revisão após vencido o prazo; gerar documento em PDF unindo artigo e revisão.

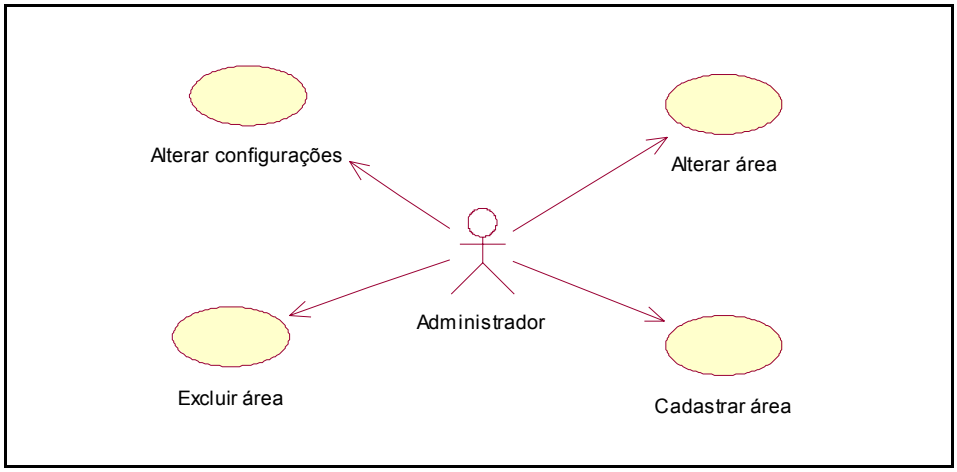

Figura 3: Diagrama de Caso de Uso do pacote Administração.

3.2 PACOTE SUBMISSÃO

Na Figura 4 está representado o comportamento do pacote submissão, em que o ator, escritor, interage com o componente por intermédio dos casos de uso: cadastrar/alterar/excluir/finalizar artigo.
No caso de uso finalizar artigo o escritor o disponibiliza para revisão. Após finalizado, o escritor não poderá mais realizar as operações de alteração e exclusão do artigo.

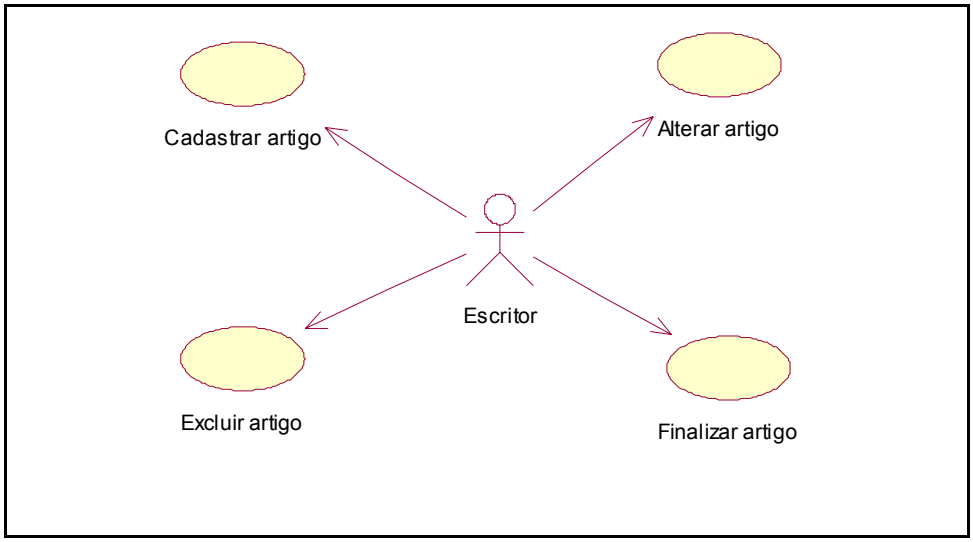

Figura 4: Diagrama de Caso de Uso do pacote submissão.

3.3 PACOTE REVISÃO

A Figura 5 mostra o comportamento do pacote revisão, em que o ator, revisor, realiza interações com o componente por meio dos casos de uso cadastrar/alterar/excluir revisão e liberar artigo.
Através do caso de uso liberar artigo o revisor poderá optar por disponibilizar o artigo no ambiente de EAD ou devolvê-lo ao escritor para realizar as correções.

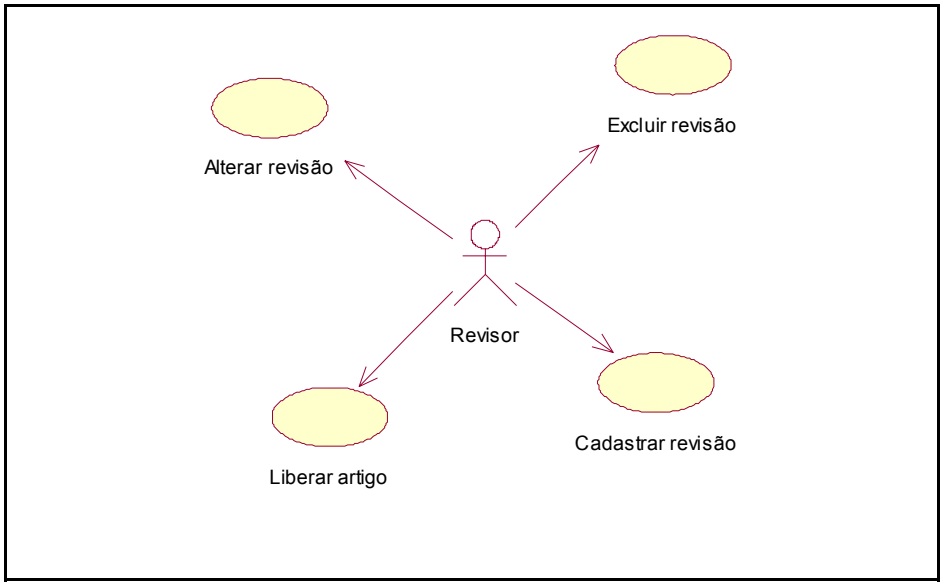

Figura 5: Diagrama de Caso de Uso do pacote revisão.

3.4 DIAGRAMA DE CLASSES

Os aspectos estáticos do *Paper Component* são abordados através do diagrama de classes, que possibilita a visualização dos objetos e métodos que compõem o sistema.

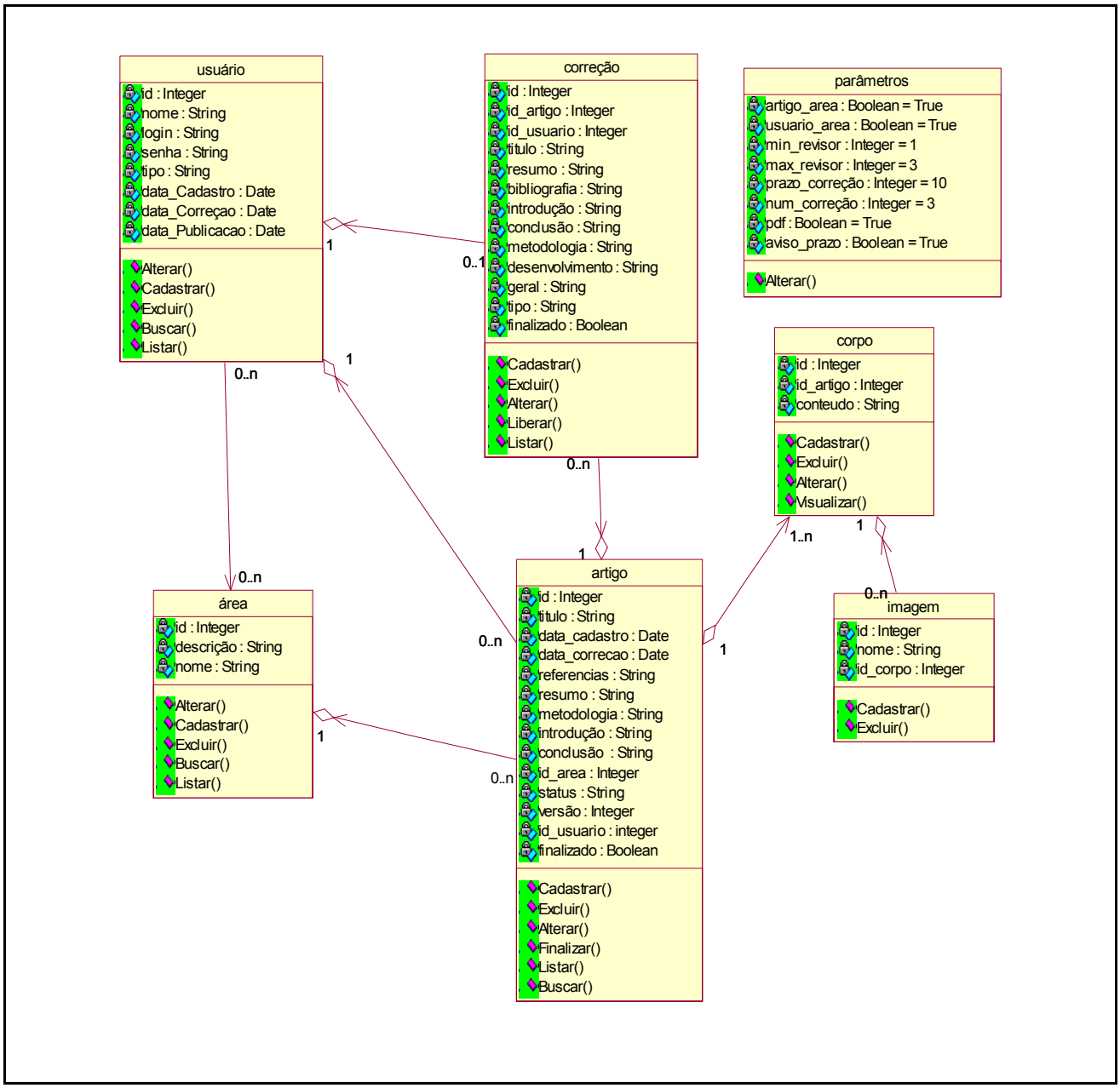

Figura 6: Diagrama de classes do componente.

O *Paper Component* compõe-se de sete classes: usuário, área, artigo, corpo, imagem, correção e parâmetros. As classes são os blocos de construções mais importantes de qualquer sistema orientado a objetos, elas descrevem um conjunto de objetos com estruturas, comportamento, relacionamento e semânticas comuns [4].

A classe parâmetros contém uma lista de atributos que são utilizados para configuração do componente. Ela não possui nenhum relacionamento, visto que seu funcionamento independe das demais classes.
O usuário se relaciona com a área através de uma cardinalidade 0..n para 0..n, o que significa que um usuário poderá ter várias áreas ou nenhuma, e que uma área poderá estar relacionada a mais de um usuário ou nenhum.
Entre as classes artigo e usuário ocorre um relação de agregação com cardinalidade de 1 para 0..n, o que estabelece que todo artigo deve ter um usuário e que um usuário pode ter vários ou nenhum artigo. A agregação representa um relacionamento estrutural entre pares, significando que essas duas classes estão conceitualmente em um mesmo nível, sem que uma seja mais importante do que a outra.
Artigo e corpo se relacionam, por agregação, através de uma cardinalidade de 1..n para 1 o que indica que o corpo pertence a um artigo e que todo artigo conterá pelo menos um corpo.
A classe imagem descreve um relacionamento de 1 para 0..n com corpo, o que indica que um corpo poderá conter zero ou mais imagens e que uma imagem só poderá estar relacionada a um corpo.
A classe correção se relaciona com as classes artigo e usuário por meio de agregação. Com a classe artigo há um relacionamento de 0..n para 1 o que indica que uma correção estará ligada a apenas um artigo e que um artigo poderá ter 0 ou mais correções. Com usuário descreve-se um relacionamento com cardinalidade 1 para 0..n, o que implica que cada correção deverá estar associada a um único usuário e que este por sua vez poderá estar relacionado a mais de uma correção.

4. ARQUITETURA PROPOSTA

Além de utilizar os diagramas de classe e de caso de uso, da UML, utilizou-se o modelo Proposto de Arquitetura em [5], visto que o mesmo especifica uma arquitetura de alto nível para tecnologia de informação de aprendizagem, instrução e sistemas de treinamento. Este modelo abrange uma grande área de sistemas, que envolve tecnologia de aprendizagem, educação, treinamento baseado em computador, instrução apoiada por computador, tutoração inteligente, entre outros.
O modelo é pedagógica e culturalmente neutro, independente de plataforma e fornece uma estrutura para compreender sistemas futuros e já existentes, promovendo a interoperabilidade e a portabilidade, identificando interfaces críticas do sistema [5].
O desenvolvimento destas arquiteturas de sistemas tem como objetivo criar descrições de alto nível para compreender determinados tipos de sistemas, seus subsistemas e suas interações com sistemas relacionados.
Uma arquitetura é uma estrutura para projetar uma escala dos sistemas sobre o tempo, para a análise, comunicação e a comparação destes sistemas. Desse modo, a arquitetura provê componentes que podem ser compartilhados entre diferentes sistemas, no nível correto da generalização. A arquitetura promove o projeto e a execução dos componentes e dos subsistemas que são reusáveis, de custo efetivo e adaptáveis, com a identificação de serviços e com interfaces de alto nível e interoperabilidade.
O modelo Proposto em [5] descreve processos, repositórios e fluxos de informação. Os processos são descritos em termos de limites, entradas, procedimentos (funcionalidades) e saídas. Repositórios são descritos pelo tipo de informação armazenado, busca, recuperação e métodos de atualização. Os fluxos são descritos em termos de conectividade (unidirecionais ou bidirecionais, conexões estáticas e dinâmicas) e o tipo de informação que flui.

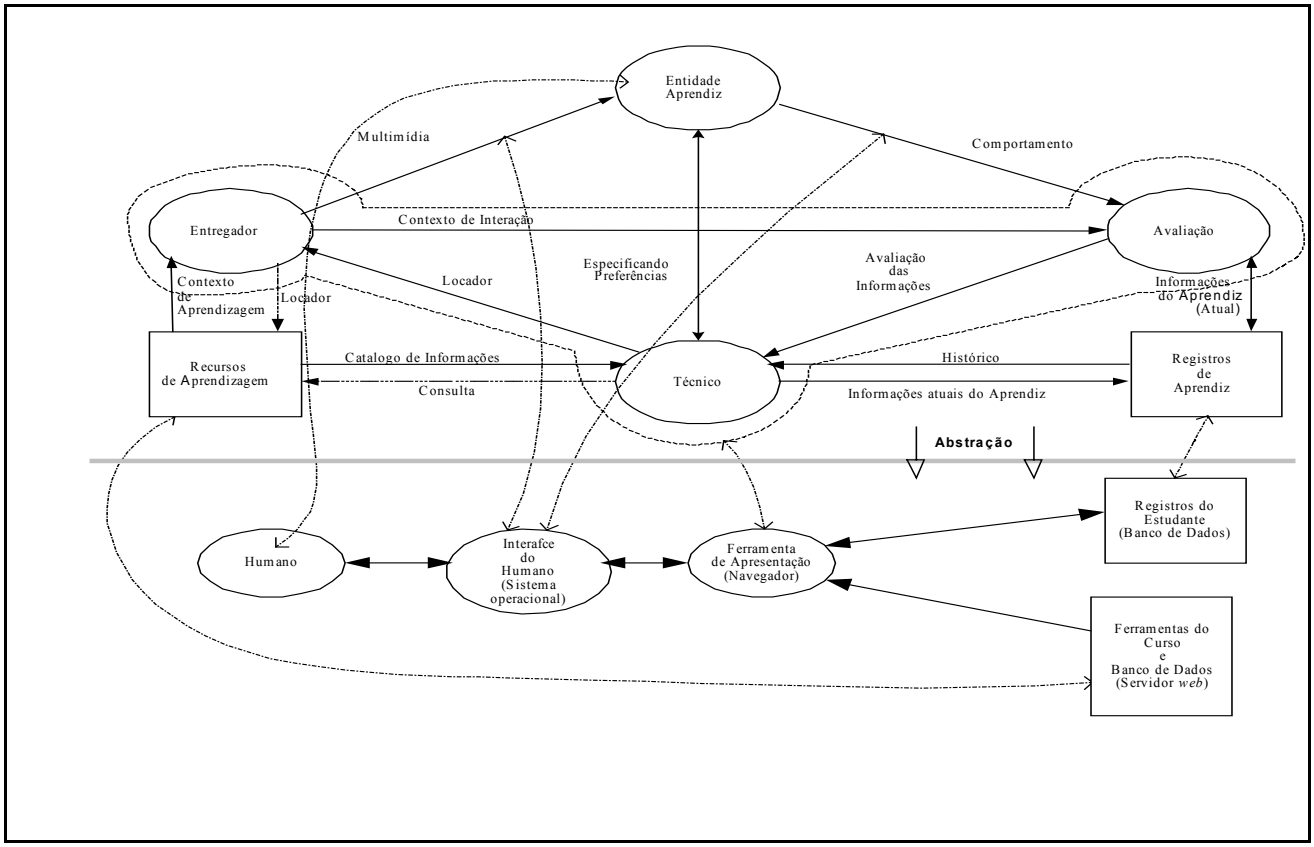

Figura 7: Arquitetura Proposta para Desenvolvimento de um Aplicação.

4.1 REPRESENTAÇÂO DO PAPER COMPONENT NA ARQUITETURA PROPOSTA

Com o intuito de visualizar melhor a representação do *Paper Component* na arquitetura, observando da Figura 7 apenas os elementos que descrevem o funcionamento do componente. Segundo a Arquitetura Proposta, identificou-se no *Paper Component* os seguintes componentes: Entidade Aprendiz, Fluxos, Técnico e Entregador.

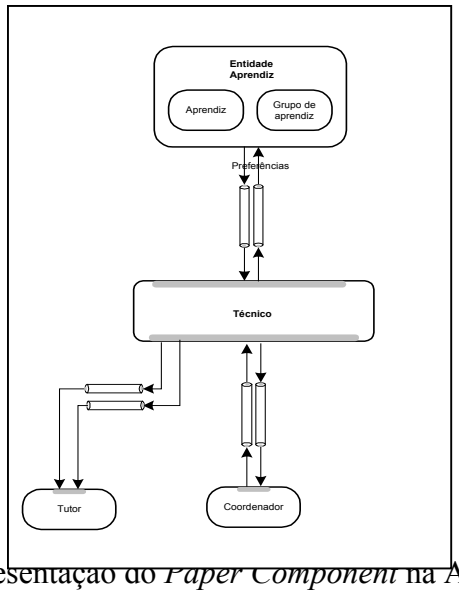

Figura 8: Representação do *Paper Component* na Arquitetura Proposta.

O usuário, o componente Aprendiz da arquitetura, interage com o *Paper Component* através de um *browser* que o possibilita acessar o ambiente de EAD, o qual é carregado através do protocolo HTTP (Hyper Text Transfer Protocol).
Para entrar no curso desejado o usuário escolhe o tipo de curso, o protocolo HTTP busca no banco de dados do Servidor, os registros de curso para o tipo de curso escolhido e carregá-os na página. Após o usuário selecionar o curso ao qual deseja acessar, o protocolo HTTP carrega a página de login, em que o usuário irá informar seu login e senha.
As informações de login e senha são variáveis, que o protocolo HTTP repassa para a linguagem. O componente técnico da arquitetura delega ao componente Administrador a responsabilidade de autenticar o usuário, que consiste em acessar o banco de dados do servidor, verificar se o usuário informado é um registro do banco de dados e se a senha informada confere com a armazenada. Após ser realizada a validação do usuário, o componente Administrador repassa para o componente Técnico o resultado da validação, que por sua vez repassa ao componente Coordenador que irá alocar os recursos segundo o tipo do usuário (administrador/escritor/revisor) e repassá-los ao Técnico. Através do fluxo de contexto o Técnico encaminha ao componente Entregador os recursos alocados.
Se o usuário logado for um escritor/revisor o componente Entregador envia para o browser, através do protocolo HTTP, uma página que permite acessar os seus artigos submetidos, no caso do escritor, e os artigos que o usuário revisor possui para analisar.
Caso o usuário seja um administrador, será enviada ao browser, uma página onde será possível, além da visualização de seus artigos submetidos e os artigos para revisar, parametrizar o componente, e cadastrar/alterar/excluir as áreas de pesquisa.

4.2 IDENTIFICAÇÃO DE PADRÕES DE PROJETO

O Padrão de Projeto (Design Pattern) é definido como o encapsulamento da descrição abstrata e estruturada de uma solução satisfatória para um problema que ocorre repetidamente dentro e um contexto, dado um conjunto de força ou restrições que atuam sobre o problema. Padrões visão melhorar, formalizar e documentar experiência e conhecimento de projeto [9].
A partir da Arquitetura Proposta em [5] foram identificados no *Paper Component* os seguintes padrões: *Composite, Façade e DataAccessObject (DAO)*.
O padrão *Composite* pode ser utilizado para compor as interfaces de comunicação com a Entidade Aprendiz e o *Façade* para compor uma interface única pois ele fornece uma interface unificada para um conjunto de interfaces num subsistema. Pode ser utilizado para compor as interfaces de comunicação com a Entidade Aprendiz. O *DAO* pode ser usado para que o componente técnico se conecte com os repositórios.

5. IMPLEMENTAÇÃO DO PROTÓTIPO

Para validar o modelo proposto implementou-se um protótipo do *Paper Component*. O protótipo foi totalmente desenvolvido utilizando-se a linguagem PHP. As motivações para a escolha dessa linguagem são: alto desempenho; interfaces para muitos sistemas diferentes de banco de dados; bibliotecas integradas para muitas tarefas comuns na Web; baixo custo; facilidade de se aprender e utilizar; portabilidade; e, disponibilidade do código fonte. O banco de dados utilizado foi o MySQL que apresenta um alto desempenho, baixo custo, facilidade de configuração e aprendizado; portabilidade e disponibilidade do código fonte [6].
A interface do componente foi desenvolvida utilizando-se HTML (Hyper Text Transfer Protocol).

A Figura 9 apresenta a tela de configurações. O componente já possui alguns valores *default,* no entanto o usuário administrador poderá alterá-los conforme as suas necessidades.

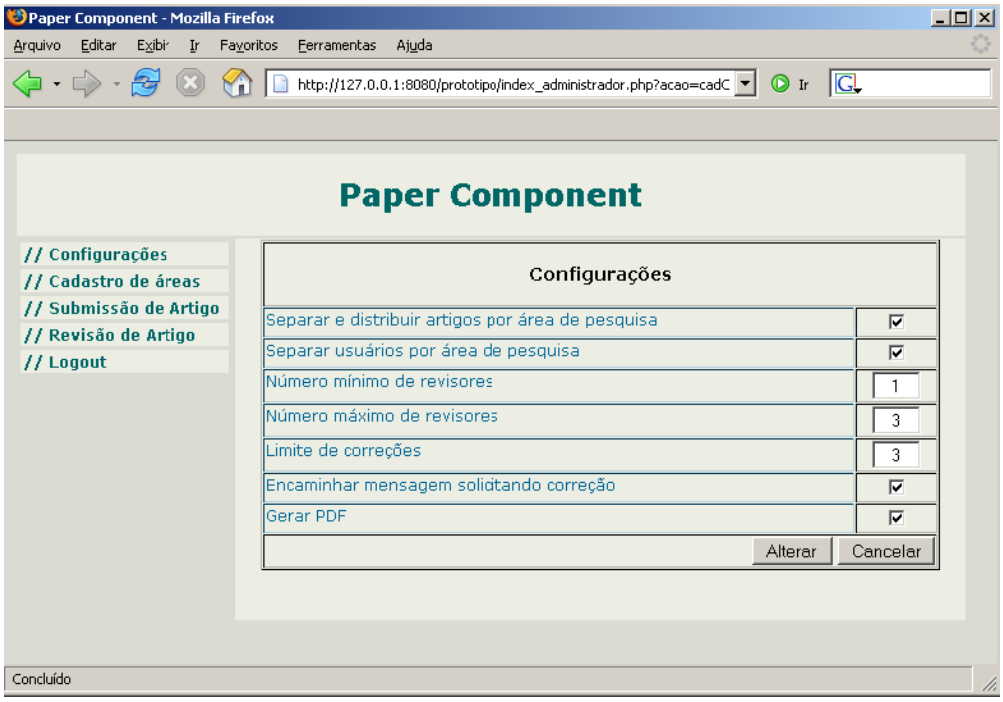

Figura 9: Interface para configuração do componente.

A Figura 10 apresenta a tela em que os usuários realizam a submissão do artigo.

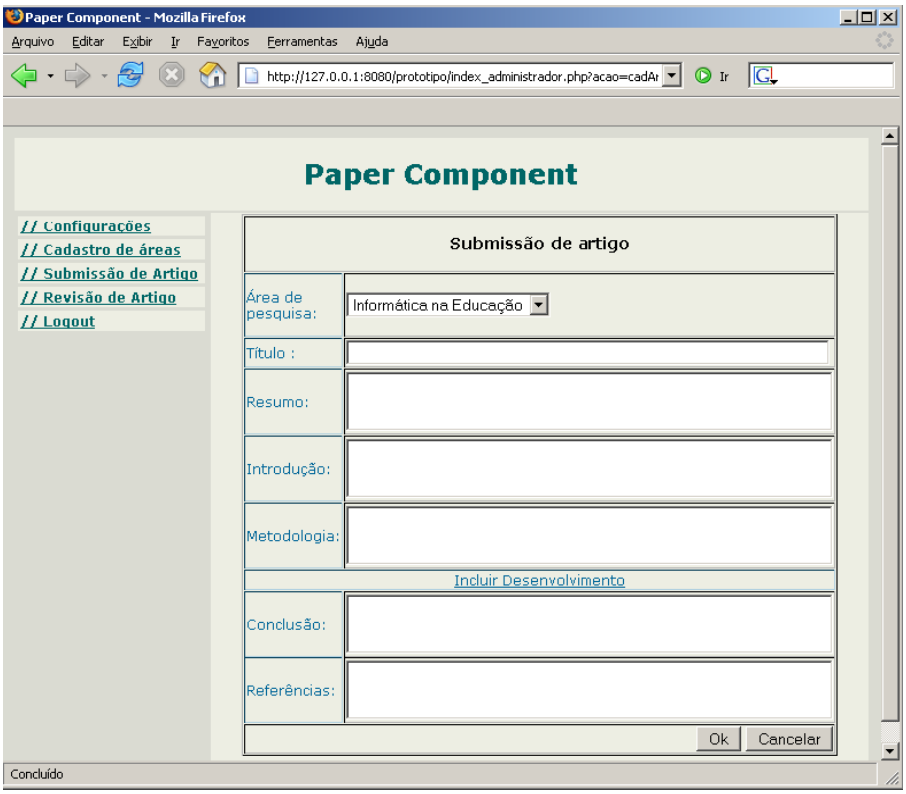

Figura 10: Interface para submissão de artigo.

6. ADAPTAÇÃO DO COMPONENTE

Normalmente componentes precisam ser adaptados para que possam satisfazer aos requisitos do sistema que serão acoplados, tendo em vista que dificilmente serão reusados da forma que foram desenvolvidos. Existem duas formas recentemente usadas para fazer esta adaptação, Colagem *(glueing)* e empacotamento *(wrapping)* [9].
Como exemplo, utiliza-ze o método de colagem para interconectar o *Paper Component*, com estrutura de interface diferente da arquitetura proposta para o ambiente de EAD. Para realizar está conexão é necessário que o componente seja adaptado para se modelar aos requisitos do sistema que será acoplado. Dessa forma deve-se criar um componente intermediário que realize a comunicação. Este componente intermediário é genericamente chamado de cola *(glue)* [9].

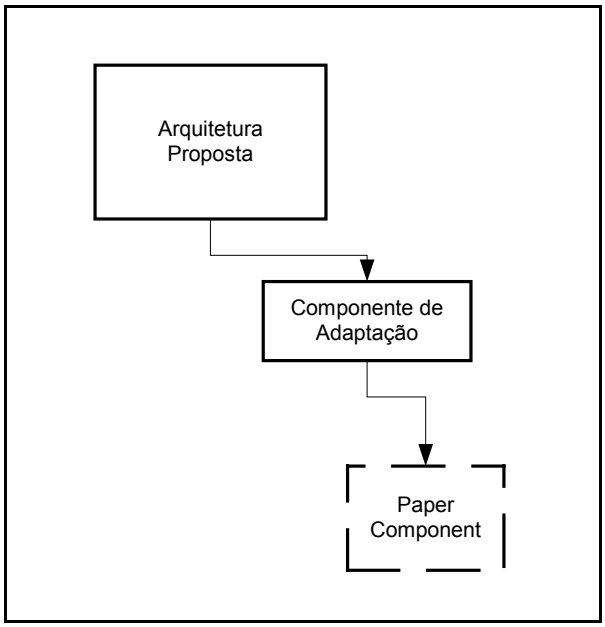

Figura 11: Adaptação de Componentes utilizando o recurso de Cola.

Supondo que o *Paper Component*, que deve ser incorporado ao ambiente de EAD, seja implementado em PHP e o ambiente de EAD, em Java. Com isto, o componente cola deve solucionar a heterogeneidade entre os outros dois componentes (que também pode envolver localização física e plataforma de execução), bem como eventuais incompatibilidades sintáticas e funcionais. Suponha-se que a questão da heterogeneidade seja resolvida com o uso de CORBA. Neste caso o componente cola mascaria o tratamento da heterogeneidade.

7. CONCLUSÃO

O processo de aprendizagem mediado por tecnologia passou a evoluir a partir do uso de computadores e principalmente da internet, com isto tem se tornado cada vez mais freqüente o uso de ambientes de EAD.
Motivados pela crescente busca por artefatos de softwares reutilizáveis, eficientes e de fácil manutenção o trabalho aborda o desenvolvimento do *Paper Component*. Enfocando a manutenibilidade e adaptabilidade o componente foi modelado segundo a UML e a arquitetura Proposta em [5], observando que é um modelo pedagógica e culturalmente neutro, independente de

plataforma, que fornece uma estrutura que promove a interoperabilidade e a portabilidade do sistema.

Com o desenvolvimento do componente apresentado neste artigo, observou-se a grande utilidade de desenvolvimento e implementação de componentes independentes. Isto torna a prática de desenvolvimento de aplicações como agregação de componentes bastante evidente e clara, inclusive mostrando como fazer a conexão entre os componentes.

Como continuidade ao trabalho, poderiam ser agregadas mais funcionalidades ao componente a fim de torná-lo mais robusto e flexível às necessidades dos usuários.

8. REFERÊNCIAS

[1] ROCHA, H. V. – **Projeto TelEduc: Pesquisa e Desenvolvimento de Tecnologia para Educação à Distância**. Em: IX Congresso Internacional de Educação a Distância da ABED (Associação Brasileira de Educação a Distância), 2002. (Trabalho Vencedor do Prêmio de Excelência ABED/EMBRATEL na categoria Pesquisa).

[2] LEMOS, M.F.R; SOUZA, M.F; SCHIRMBEMCK, F.R.G. – **Educação a Distância: Uso do Teleduz – Uma realidade no Aprender a Aprender**. Disponível por WWW em : http://www.abed.org.br/congresso2004/por/htm/094-TC-C3.htm. Consultado em 20/04/2005.

[3] HUZITA, E. H. M.; GIMENES, I. M. S. – **Desenvolvimento Baseado em Componentes.** Editora Ciência Moderna, 2005.

[4] BOOCH, G; RUMBAUGH, J; JACOBSON, I. – **UML – Guia do Usuário**. Editora Campus, 2000.

[5] VASCONCELOS, J.R; RICARTE, I. L.; MARCHI, G. D.; GATTO, R. A. – **Uma Abordagem de Arquitetura Estilizada para Software Educacional.** Em: III – Workshop de Tecnología Informática Aplicada en Educación, 2004.

[6] WELLING, L; THOMSON, L. – **PHP e MySQL Desenvolvimento Web.** Editora Campus, 2003.

[7] PFISTER, Cuno. – **A Case Study using BlackBox Components**. Disponível por WWW em: http://www.oberon.ch/docu/case_study/ index.html. Consultado em 17/08/1998.

[8] SILVA, R. P. – **Suporte ao desenvolvimento e uso de Frameworks e componentes.** Doutorado em Ciência da Computação – UFRGS, 2000.

[9] VASCONCELOS, J.R; RICARTE, I. L.; MARCHI, G. D.; GATTO, R. A. – **Padrões de Projeto (Design Patterns).** Relatório do projeto de conclusão de curso, 2004.