UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
DEPARTAMENTO DE INFORMÁTICA
MESTRADO EM INFORMÁTICA

PEDRO DAVID NETTO SILVEIRA

O USO DE TEMPLATES PARA AGILIZAR A CUSTOMIZAÇÃO DE AMBIENTES COLABORATIVOS

VITÓRIA, 2012

PEDRO DAVID NETTO SILVEIRA

O USO DE TEMPLATES PARA AGILIZAR A CUSTOMIZAÇÃO DE AMBIENTES COLABORATIVOS

Dissertação submetida ao Programa de Pós-Graduação em Informática da Universidade Federal do Espírito Santo como requisito parcial para a obtenção do grau de Mestre em Informática.

VITÓRIA, 2012

Dados Internacionais de Catalogação-na-publicação (CIP)
(Biblioteca Central da Universidade Federal do Espírito Santo, ES, Brasil)

Silveira, Pedro David Netto, 1985-
S587u O uso de templates para agilizar a customização de ambientes colaborativos / Pedro David Netto Silveira. – 2012.
127 f. : il.

Orientador: Crediné Silva de Menezes.
Coorientador: Davidson Cury.
Dissertação (Mestrado em Informática) – Universidade Federal do Espírito Santo, Centro Tecnológico.

1. Interface gráfica com o usuário (Sistemas de computação). 2. Ambientes virtuais compartilhados. I. Menezes, Crediné Silva de, 1952-. II. Cury, Davidson. III. Universidade Federal do Espírito Santo. Centro Tecnológico. IV. Título.

CDU: 004

PEDRO DAVID NETTO SILVEIRA

O USO DE TEMPLATES PARA AGILIZAR A CUSTOMIZAÇÃO DE AMBIENTES COLABORATIVOS

Dissertação submetida ao Programa de Pós-Graduação em Informática da Universidade Federal do Espírito Santo como requisito parcial para a obtenção do grau de Mestre em Informática.

Aprovada em 16 de Abril de 2012.

COMISSÃO EXAMINADORA

_____

Prof. Dr. Crediné Silva Menezes
Universidade Federal do Espírito Santo (UFES)
(Orientador)

_____

Prof. Dr. Davidson Cury
Universidade Federal do Espírito Santo (UFES)
(Co-Orientador)

_____

Prof. Dr. Orivaldo Lira Tavares
Universidade Federal do Espírito Santo (UFES)
(Membro Interno)

_____

Prof. Dr. Alberto Nogueira de Castro Junior
Universidade Federal (UFAM)
(Membro Externo)

VITÓRIA, 2012

DEDICATÓRIA

Para minha família

AGRADECIMENTOS

Agradeço a Deus, meu Senhor, em primeiro lugar, sempre.

Agradeço ao meu mentor e amigo, Prof. Crediné, pelo compartilhamento de ideias, pelo incentivo e pelas palavras certas nos momentos certos.

Ao membro externo da banca examinadora, Prof. Alberto, que aceitou o convite, dispondo tempo e conhecimento para avaliação dessa dissertação.

Aos membros internos da banca examinadora, Prof. Orivaldo e Prof. Dede, pelas boas aulas, pelos auxílios na construção do meu conhecimento e pela descontração.

À FAPES, pelo provimento da bolsa.

Aos colegas do LIED, Everton, Maikson, Alexandre, Ernani, Thalita, Rafalski, Otávio, Halysson, Ramon e Gazela. Jamais me esquecerei de vocês.

Aos amigos pessoais, Bill, Alemão e Titia. Amo vocês como irmãos e sei que posso contar com sua ajuda sempre. Meu carinho por vocês é imenso.

À galera da graduação, Carlos, Kelly, Juliana, Marcellus e tantos outros. Mesmo na minha tese de doutorado não me esqueceria de vocês. Vocês contribuíram muito para que eu chegasse aqui.

À minha querida irmã, Carol, cunhado, Tárik e azeitoninha, minha sobrinha que está na barriguinha da mamãe. Os amo do fundo do coração. Obrigado pelas conversas descontraídas, pelo carinho e pela alegria. Aliviaram muito o Estresse.

Aos meus pais, Iraldo e Márcia, razão da minha inspiração. Os maiores incentivadores e grandes amigos. Meu amor por vocês existirá enquanto eu viver. Obrigado por tudo.

À minha esposa e amor da minha vida, Flavinha. Isso tudo só foi possível graças à sua paciência, devoção, carinho e ajuda. Devo esta vitória a você.

Mas em todas estas coisas somos mais do que vencedores, por aquele que nos amou.

Romanos 8:37

RESUMO

A crescente utilização dos ambientes virtuais colaborativos nas mais variadas formas destaca lacunas nos suportes tecnológicos atuais no que diz respeito à customização do ambiente e de sua interface. Esse fato incita a criação de novas propostas que atendam às necessidades ou preferências dos usuários de espaços virtuais. Este trabalho apresenta uma arquitetura de software que apoia a autoria na criação e edição de interfaces visuais dos espaços virtuais desenvolvidos a partir do MOrFEu, favorecendo a personalização dos ambientes e consequentemente gerando sua melhor adaptação aos estilos cognitivos de seus usuários. Os *templates*, instâncias de interfaces gráficas, são criados individualmente ou em conjunto. Seus usuários são ainda beneficiados com a possibilidade de sua reutilização e customização.

Palavras-chave: *Template*, Interface Gráfica, Ambientes Virtuais, Customização, Flexibilidade.

ABSTRACT

The increasing use of collaborative virtual environments in many forms highlights gaps in current technological support with respect to customization of the environment and its interface. This fact urges the creation of new proposes to assist with the needs or preferences of virtual spaces participants. This work presents a software architecture that supports the authoring and editing in the creation of virtual spaces visual interfaces developed from MOrFEu, providing ways of customizing environments and, consequently, generating better adaptation to the cognitive styles of their users. The templates, instances of graphical user interfaces are created individually or with other users. Users are also benefited with the possibility of reuse and customization.

Keywords: Template, Graphic Interface, Virtual Environments, Customization, Flexibility

SUMÁRIO DE FIGURAS

SUMÁRIO DE TABELAS

SUMÁRIO

CAPÍTULO 1 INTRODUÇÃO

Nos últimos tempos, acompanhamos um contínuo crescimento da utilização da internet como meio de comunicação. Dentro desse contexto, grupos de usuários se formam a fim de compartilhar informações, formar opiniões, adquirir conhecimento e colaborar, à distância, das mais diversas formas.

Com o advento da web 2.0, no contexto de mediação de conteúdo, estão inseridas variadas plataformas e ferramentas que apoiam as pessoas em suas atividades em grupo. Uma instância desse contexto está na realização da educação à distância (EaD) utilizando os conhecidos Ambientes Virtuais.

Pode-se afirmar que um ambiente virtual é um espaço no qual seres humanos e objetos interagem, potencializando assim, a construção de conhecimentos [SANTOS, 2002]. Para esse fim, vivenciamos o surgimento de inúmeros projetos nos últimos anos. Contudo, como afirmado em [BLACK et al. 2007], ambientes virtuais existentes são muito parecidos e padronizados, ou seja, são ferramentas essencialmente desenvolvidas para suporte de atividades previamente existentes.

Nota-se hoje, uma grande carência na exploração da organização flexível de espaços virtuais. Uma das maiores causas de frustração dos educadores é justamente a ausência de ambientes computacionais adaptáveis ou adequados a diferentes propostas de trabalho que exigem diferentes recursos.

Em [MENEZES et al., 2008] foram analisadas as dificuldades encontradas na prática de pedagogias abertas, originadas pelas concepções tecnológicas atuais. Essas dificuldades estão ligadas principalmente à falta de flexibilidade na utilização dos ambientes, no que diz respeito a (i) descrição do espaço virtual, (ii) os momentos nos quais os participantes deverão interagir e (iii) a manipulação da interface com o usuário. Neste trabalho abordaremos principalmente as dificuldades mencionadas em (ii) e (iii).

Uma interface com o usuário, amigável e personalizada, em ambientes de aprendizagem é muito importante para o desenvolvimento do interesse do participante em relação ao ambiente. Isso pode ser afirmado com base em [ARRIGO, 2008], que destaca a importância da relação entre a manipulação da interface visual do espaço virtual e as preferências ou necessidades que as pessoas têm.

Nesse contexto emerge o estudo sobre as interfaces adaptativas, que estimula o desenvolvimento de sistemas capazes de promover a adaptação de conteúdos e recursos hipermídia, vindos de qualquer fonte (bancos de dados, Internet, serviços etc.) e apresentados em qualquer formato (texto, áudio, vídeo ou suas combinações) ao perfil ou modelo de seus usuários. A Hipermídia adaptativa encontra aplicação direta, por exemplo, em tecnologias atuantes sobre educação, sistemas de informação, comércio eletrônico, lazer, necessidades especiais etc.

Existem diversos trabalhos no domínio das interfaces adaptativas tais como: [UJITA, 1992], [BODART, 1994] e [FOLEY, 1991], que desenvolveram ferramentas para controlar a adaptação das interfaces, usando técnicas de inteligência artificial e modelos cognitivos humanos. No entanto, esses trabalhos têm sido insuficientes e apresentam lacunas com relação a usabilidade de interface. Segundo [FURTADO, 1999], os problemas acontecem parcialmente porque recomendações ergonômicas são raramente levadas em consideração. Assim, as adaptações efetuadas nem sempre são de interesse do usuário.

Quando falamos em interfaces de ambientes virtuais de aprendizagem (AVA) que são usados em larga escala e concomitantemente por diferentes pessoas, obter um alto grau de interesse dos participantes do ambiente se torna uma tarefa muito difícil, e nem sempre é possível em sua totalidade.

Uma das preocupações em um projeto de interface com o usuário é permitir maior flexibilidade no acesso à informação, para que usuários com diferentes necessidades sejam atendidos [SCHIMIGUEL et al., 2005]. Assim, um projeto

de interface deve antecipar-se a restrições tecnológicas e a necessidades específicas das pessoas, procurando minimizar a carga cognitiva para execução de tarefas, diminuir as possibilidades de erros ou fracassos e motivar o uso dessa interface [SALES e CYBIS, 2003].

Considerando os recursos atuais e as facilidades no meio computacional, é esperado que um cidadão qualquer possa aprender e aplicar roteiros para construir um ícone, um curso para EaD, uma *home page* ou até mesmo um AVA em forma de *website*.

Se as pessoas tiverem em mãos ferramental que proporcione personalização de interface de seus espaços virtuais, estarão favorecendo a si com mecanismos visuais adequados às tarefas que devem ser realizadas. Isso deverá possibilitar uma maior aceitação, o que vale dizer, que deve atender às demandas do usuário inserido no contexto do AVA, de forma que melhor proporcione satisfação, elimine eventuais dúvidas na percepção visual e incremente o grau de retenção das informações por tempo mais longo.

Nos últimos anos, nosso grupo realiza uma pesquisa, com participação de pesquisadores da UFES, UFAM e UFRGS, que busca a concepção de uma tecnologia inovadora, um meta-ambiente para o desenvolvimento de AVAs. Esta pesquisa, com diferentes frentes de trabalho, se constitui no Projeto **MOrFEu – Multi Organizador Flexível de Espaços Virtuais** [MENEZES et al., 2008], que busca a definição de modelos, estratégias e implementações para dar suporte à construção de espaços virtuais colaborativos.

Neste trabalho, apresentaremos uma proposta de arquitetura de software capaz de apoiar a elaboração da interface dos ambientes virtuais colaborativos construídos a partir do MOrFEu, isto é, um editor que propicie construção/edição dos componentes da camada de apresentação (*layout*, *design*, navegação etc) do espaço virtual criado no meta-ambiente, através do suporte computacional no favorecimento às atividades de autoria individual ou coletiva, levando em consideração a manipulação direta e as interfaces exploráveis.

Nas próximas sessões, falaremos o que nos motivou a realizar essa pesquisa, os objetivos que alcançamos no decorrer do desenvolvimento deste trabalho, a metodologia usada e o restante do conteúdo que abordaremos aqui.

1.1 MOTIVAÇÃO

O que temos visto nos ambientes virtuais atuais é uma imposição da utilização da estrutura e interface que o espaço fornece. Contudo, existe uma forte demanda ao atendimento particular das necessidades ou das preferências dos participantes em relação ao ambiente. Nesse contexto, os espaços virtuais para aprendizagem, muitas vezes não podem ser adaptados para suprir essa demanda.

A principal motivação para a realização dessa pesquisa envolve realizar as expectativas dos usuários do MOrFEu no que tange o desenvolvimento da interface dos espaços virtuais construídos por eles, isto é, fazer com que o indivíduo não se frustre ao se deparar com uma interface que não lhe agrade ou que o desmotive, como no caso de pessoas com necessidades especiais como o discromatopsia ou daltonismo. O MOrFEu apresenta algumas estruturas básicas que o compõem, incluindo estruturas referentes às interfaces visuais. A segunda grande motivação está na existência de um mundo inexplorado sobre o estudo da camada de apresentação dos ambientes criados no MOrFEu.

1.2 MORFEU

A criação de ambientes virtuais flexíveis é o maior resultado que o projeto MOrFEu se propõe a obter, e para essa realização, se faz necessária uma modularização da concepção, organizando o projeto em três elementos que descrevem a estrutura, o conteúdo e a aparência do ambiente criado. Esses elementos são respectivamente o Veículo de Comunicação (VCom), a Unidade de Produção Intelectual (UPI) e o *Template*.

Cada elemento do MOrFEu (UPI, VCom e *Template*) apresenta um editor responsável pela criação/edição dos elementos. A Figura 1.1 exemplifica o núcleo e os componentes que podem ser inseridos em seu contexto. Dos elementos presentes na figura, o **Editor de Templates** (Capítulo 4) é o fruto deste trabalho. Todos os outros elementos estão detalhados no Capítulo 3.

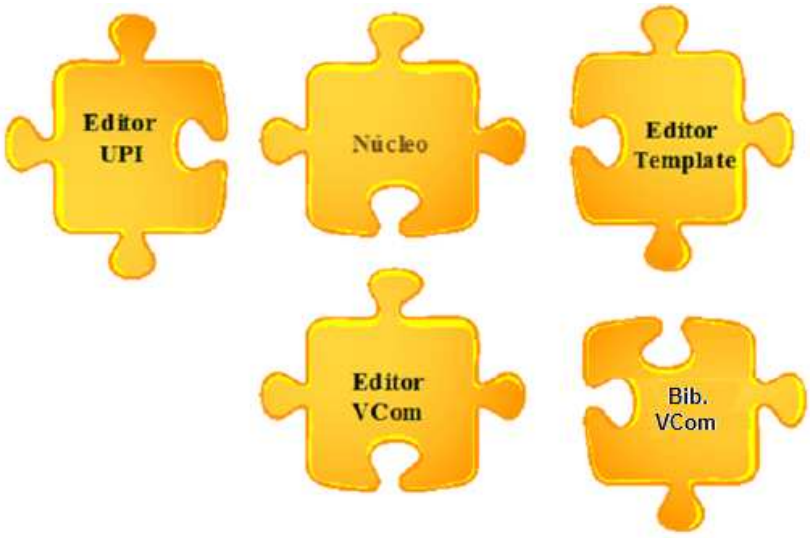


Figura 1.1: Visão geral sobre os editores do MOrFEu. (Imagem estendida [VIEIRA, 2011])

Alunos de graduação do curso de Ciência da Computação da Universidade Federal do Espírito Santo, têm desenvolvido alguns trabalhos relacionados ao MOrFEu como projetos de conclusão de curso. Um deles é a biblioteca de VCom que se liga aos Editores de Templates e VCom (Figura 1.1). Com essa biblioteca os participantes do ambiente poderão realizar buscas em uma base de dados a fim de construir seus próprios espaços virtuais, partindo de um VCom previamente instanciado.

1.3 OBJETIVOS

O principal objetivo da arquitetura proposta nesta dissertação está em explorar e caracterizar os requisitos de interface que devem compor a camada de apresentação do MOrFEu. Isso implica em desenvolver uma primeira versão do editor inerente às interfaces, cujo produto final damos o nome de ***Templates***.

O Editor de *Templates* do MOrFEu é desenvolvido para atuar sobre todos os componentes de interface do espaço virtual, e isso significa suprir as seguintes características:

i. Não podemos esperar que os indivíduos operantes no sistema estejam aptos a adaptar a interface às suas necessidades ou preferências no nível de codificação. Devemos fornecer a eles outros meios para a construção da aparência do ambiente.
ii. Prover total separação entre conteúdo e interface visual. O que facilita a recriação da aparência do ambiente em cada edição;
iii. Fornecer opção para alterações básicas sobre as produções como, por exemplo: cor e tamanho de fonte, cor de plano de fundo, espaçamento do texto, margens, segregação de texto e imagem, tamanho da imagem etc.;
iv. Permitir aos participantes do ambiente determinar a disposição dos elementos na página, modificando a visualização dos componentes da estrutura;
v. Permitir ao usuário definir o modo como navegar sobre os espaços virtuais, isto é, definir a transição entre as páginas do ambiente;
vi. Executar regras de acessos de escrita e leitura sobre conteúdo, determinando diferentes visões para cada usuário.

1.4 METODOLOGIA E HISTÓRICO DO DESENVOLVIMENTO DO TRABALHO

A presente dissertação passou por algumas etapas envolvendo especificação de requisitos, estudos de trabalhos correlatos, provas de conceito e estudos de caso. As descrições a seguir mostram como se deu a realização de cada etapa.

Inicialmente realizou-se a pesquisa bibliográfica que foi dividida em três estudos: Primeiro foi feita uma revisão sobre trabalhos relacionados à padronização e modelagem da camada de apresentação da web. No segundo estudo buscamos entender como determinar a melhor forma de construção de

interface na web por meio de alguns estudos recentes. No terceiro estudo foram analisadas as tecnologias atuais e comuns que atuam sobre interfaces com usuário.

O primeiro estudo, da primeira etapa deste trabalho, foi realizado a fim de contextualizar adequadamente a proposta, fazendo uma revisão bibliográfica sobre o que existe no meio acadêmico relacionado à estruturação de interface com usuário voltada para ambientes colaborativos, incluindo participantes comuns e participantes com necessidades especiais.

No segundo estudo foram abordadas as tecnologias atuais utilizadas para desenhar interfaces e padronizar aparência de documentos da web, como por exemplo: *Cascading Style Sheets (CSS), eXtensible Markup Language (XML), eXtensible Stylesheet Language (XSL), eXtensible Stylesheet Language Transformations (XSLT)* dentre outras. Além disso, também foram abordados ambientes virtuais correlatos de aprendizagem. As etapas seguintes constituem os passos realizados para a concepção do Editor de *templates* do MOrFEu. Em cada uma das próximas etapas foi obtido um módulo do editor, responsável por delimitar e fornecer mecanismos para criar/editar as estruturas de interface identificadas na etapa anterior.

Na segunda e terceira etapas foram concebidos os dois primeiros módulos do editor de templates. O primeiro trata-se de uma parte do editor que proporciona ao usuário a funcionalidade de determinar a disposição dos elementos da página na interface.O segundo permite ao usuário descrever a aparência figurativa (cor, tamanho, borda etc.) de cada elemento. Ao conceito de disposição dos elementos, usamos o nome genérico de ***layout***, e ao conceito de descrição de aparência figurativa, demos o nome de ***design***.

A quarta etapa foi centrada na construção de mecanismos de transição de páginas, isto é, a **navegação** entre as páginas do ambiente. Um estudo paralelo foi introduzido nesse período, tendo a ver com a avaliação de usabilidade de interfaces web sobre transição de páginas.

A quinta e última etapa constituiu na estruturação de aparatos que apoiam o MOrFEu nas definições de permissões sobre leitura e escrita de conteúdo nos espaços virtuais. Cada usuário dos espaços virtuais criados no MOrFEu deverá enxergar o sistema de forma pessoal, e isso inclui a interface personalizada, o conteúdo disponibilizado a ele e as ações que ele pode desempenhar sobre o conteúdo. A esse conceito, damos o nome de ***visões*** sobre o ambiente.

Utilizando essa metodologia obtivemos resultados esperados tais como estudos detalhados sobre produção de *templates*, uma boa revisão bibliográfica, uma especificação do software e uma arquitetura para a resolução do problema.

1.5 ORGANIZAÇÃO DA DISSERTAÇÃO

- *Capítulo 1 (Introdução)* - Apresenta uma breve introdução sobre o tema da dissertação e onde se contextualiza o trabalho, apresenta o problema e objetivos da pesquisa e a metodologia utilizada.

- *Capítulo 2 (Fundamentação Teórica)* - Apresenta o referencial teórico sobre como se dá a organização dos elementos de interface em *sites* de conteúdo e apresenta os trabalhos correlatos.

- *Capítulo 3 (MOrFEu e Templates)* - Apresenta o MOrFEu, seus componentes e a definição mais precisa dos *Templates*.

- *Capítulo 4 (Proposta para Tratamento de Templates)* - Apresenta a proposta que desenvolvemos para tratamento dos *Templates*, suas características e a modelagem conceitual do projeto.

- *Capítulo 5 (Uma Implementação)* - Apresenta a implementação da primeira versão do Editor de Templates, com exemplos práticos de criação de interfaces para espaços virtuais.

- Capítulo 6 (Exemplos de Uso) - Apresenta um exemplo de uso que tem como base o espaço virtual intitulado “Debate de Teses”.

- *Capítulo 7 (Considerações Finais)* - Apresenta as considerações finais, que incluem: experiências adquiridas, análise dos resultados obtidos, limitações da solução e trabalhos futuros.

CAPÍTULO 2 REFERENCIAL TEÓRICO E TRABALHOS CORRELATOS

Neste capítulo contextualizaremos o trabalho, abordaremos o que se tem visto sobre modelagem de hipermídias, o que se tem feito a fim de facilitar o trabalho das pessoas na manutenção de seus espaços na web e a importância da elaboração do espaço e interface pelo próprio indivíduo que o irá usar.

2.1 CONTEXTUALIZAÇÃO

[ZEVE, 2003] afirma que nos últimos anos notou-se um aumento considerável de trabalhos colaborativos à distância utilizando como suporte a web, chegando-se a conclusão de que é cada vez maior o número de pessoas envolvidas no processo de autoria desses cursos, incluindo seus próprios participantes.

O MOrFEu busca flexibilizar a construção de espaços virtuais colaborativos, justamente valorizando o processo de autoria, de forma a estar presente em todas as camadas dos ambientes criados a partir dele: estrutura do espaço (veículo de comunicação), conteúdo (UPI) e aparência (template).

Este trabalho está inserido no contexto da produção de interfaces gráficas flexíveis para os ambientes virtuais produzidos no MOrFEu. E, para realizar o projeto é necessária uma revisão bibliográfica que dê suporte ao desenvolvimento do trabalho.

Na seção 2.2 estudamos as principais abordagens sobre modelagem de hipermídias orientadas a objeto e extraímos delas características que compõem o projeto de templates. Essa caracterização será detalhada na Seção 2.2.6.

Na Seção 2.3 enunciamos a principal característica que queremos nos templates, a partir do estudo das modelagens, a fim de apoiar a flexibilização da camada de apresentação dos veículos de comunicação. Posteriormente são detalhados os impactos da usabilidade nos ambientes virtuais, e os obstáculos que precisamos transpor para garantir bons padrões de interação dos usuários com os ambientes.

Finalmente na Seção 2.5 está detalhado o estudo que justifica o encargo de colocar nas mãos dos usuários, a condição de realizar autoria na apresentação dos ambientes virtuais nos quais eles são integrantes.

2.2 ABORDAGENS DE MODELAGEM DE HIPERMIDIAS

Recentemente, diferentes abordagens para modelar *hypermedia* ou aplicações web têm emergido. Entre elas, as contribuições mais significantes que mencionamos, são: *Object Oriented Hypermedia Design Model* (OOHDM [SCHWABE et al, 1996]), *Web Modeling Language* (WebML [CERI et al, 2003]), *UML-based Web Engineering approach* (UWE [KOCH et al, 2001]), *Object-Oriented Web-Solutions Modelling* (OOWS [PASTOR et al, 2003]), e *Object-Oriented Hypermedia Method* (OO-H [GOMES & CACHERO, 2003]). Detalhamos algumas delas, a seguir.

2.2.1 OOHDM [SCHWABE et al, 1996]

Object Oriented Hypermedia Design Model – OOHDM (Metodologia de Projeto Hipermídia Orientado a Objetos) é uma metodologia para desenvolvimento de aplicações web que divide o processo de construção da aplicação hipermídia em quatro fases: Modelagem Conceitual, Projeto de Navegação, Projeto da Interface Abstrata e Implementação.

A modelagem conceitual em OOHDM é feita utilizando princípios conhecidos da modelagem orientada a objetos e tem como produto final um esquema de classes e objetos construído a partir de sub-sistemas, classes e relações.

Em uma aplicação web usada por um conjunto de usuários, tentando realizar um determinado conjunto de tarefas, é necessário, em geral, organizar as informações representadas no modelo conceitual. Em OOHDM, isto é alcançado por meio da definição de um modelo de navegação que é uma visão (no sentido de bases de dados) do modelo conceitual. Isso reflete o ponto de vista que uma das principais características distintivas de aplicações hipermídia que é a noção de navegação.

Uma vez que a estrutura de navegação for definida, ela deve estar disponível para o usuário através da interface do aplicativo, que é descrita pelo modelo de interface abstrata. Nessa etapa serão definidos: (i) quais objetos de interface que o usuário deverá perceber e, em particular, a maneira que diferentes objetos navegacionais devem aparecer, (ii) quais objetos de interface ativarão a navegação, (iiii) a forma como objetos de interface multimídia serão sincronizados e (iv) quais transformações de interface deverão acontecer.

Em OOHDM são usados *Abstract Data Views* (ADV) para especificar o modelo de interface abstrata. ADVs são usados para especificar formalmente a separação entre a interface com o usuário e os componentes de sistema de software e também para oferecer um método de projeto independente de implementação. Ainda, em OOHDM, existe o conceito de *ADVcharts*, que são a expressão interfacial dos diagramas de navegação, isto é, eles expressam transformações no nível da interface de usuário e seu impacto nos objetos navegacionais.

A implementação de uma aplicação hipermídia de modo que seja usável não é tarefa simples. Nessa fase, é necessário definir os objetos de interface de acordo com a especificação da interface abstrata, implementar transformações da forma como foram definidas nos ADVcharts e fornecer suporte para a navegação através da rede hipermídia.

Os benefícios de utilizar OOHDM dentre outras coisas, envolve: (i) aplicações projetadas e construídas em torno de objetos tendem a ser mais robustas e fáceis de modificar, tanto pelo uso de polimorfismo e herança como por

composição; (ii) construir aplicações reutilizando componentes existentes é altamente viável quando os componentes são descritos como objetos; (iii) existem poderosos formalismos para especificar a estrutura, o comportamento e as relações dos objetos que podem ser adaptados ao campo da hipermídia.

2.2.2 WebML [CERI et al, 2003]

WebML (*Web Modeling Language*) busca permitir aos *design*ers expressar as características principais de um site a um nível elevado, sem se comprometer com detalhes de pormenores arquiteturais. Os conceitos de WebML estão associados a uma representação gráfica intuitiva, que pode ser facilmente suportada por ferramentas CASE e efetivamente comunicada aos membros não-técnicos da equipe de desenvolvimento da hipermídia (por exemplo, com os *design*ers gráficos e produtores de conteúdo). A especificação de um site em WebML consiste em quatro perspectivas ortogonais: *Structural Model* (Modelo Estrutural)*, Hypertext Model* (Modelo de Hipertexto)*, Presentation Model* (Modelo de Apresentação)*, Personalization Model* (Modelo de Personalização).

O Modelo estrutural expressa o conteúdo de dados do site, em termos de entidades e relacionamentos relevantes. WebML não propõe uma linguagem de modelagem de dados alternativa, mas é compatível com as notações clássicas, como entidade e relacionamento do modelo, o ODMG orientada a objeto modelo e diagramas de classe UML.

O Modelo de Hipertexto descreve um ou mais hipertextos que podem ser publicados no site. Cada diferente hipertexto define uma então chamada “visão do site”. Descrições das visões do site, por sua vez, consistem em dois sub-modelos: Modelo de Composição e Modelo de Navegação.

O modelo de composição especifica quais as páginas que compõem o hipertexto e quais unidades de conteúdo compõem uma página. Os tipos de unidades de conteúdo que podem ser usados para compor páginas, são: dados, multi-dados, índice, filtro, e as unidades scroller diretas. Unidades de

dados são utilizadas para publicar as informações de um único objeto (por exemplo, um álbum de música), enquanto que os demais tipos de unidades representam formas alternativas de navegar em um conjunto de objetos (por exemplo, o conjunto de faixas de um álbum).

O modelo de navegação busca expressar como as páginas e unidades de conteúdo estão ligadas para formar o hipertexto. As Ligações podem ser não-contextuais, quando elas conectam páginas semanticamente independentes (por exemplo, a página de um artista para a página inicial do site), ou contextuais, quando o conteúdo da unidade de destino do link depende do conteúdo da fonte.

O Modelo de apresentação expressa a aparência gráfica das páginas por meio de uma sintaxe XML abstrata. As especificações da apresentação podem ser específicas da página ou genéricas. No primeiro caso, a apresentação de uma página é especificada baseada nas referências explícitas ao conteúdo da página; no segundo caso, a apresentação é baseada em modelos pré-definidos independente do conteúdo da página.

No Modelo de Personalização existem as entidades pré-definidas usuário e grupos de usuários. As características dessas entidades podem ser usadas para armazenar conteúdo específico do grupo ou do indivíduo, como sugestões de compras, lista de favoritos além dos recursos para personalização gráfica. Personalização é a definição de conteúdo ou estilo de apresentação com base em dados do perfil de usuário. Em WebML, unidades, páginas, seus estilos de apresentação e pontos de vista do site podem ser definidos de forma a facilitar a navegação dos usuários ou grupo de usuários dentro da hipermídia.

Algumas, dentre outras características do WebML, incluem: (i) **Derivação**: É o processo de adição de informações redundantes com o esquema de estrutura, a fim de aumentar a sua expressividade. (ii) **Modelagem de usuário**: A fim de apoiar a personalização, WebML inclui uma noção explícita de grupo e usuário, (iii) **Personalização declarativa**: O *design*er define conceitos derivados, cuja definição depende de informações específicas do usuário. (iv) **Personalização**

procedural: WebML inclui uma sintaxe XML para escrever regras de negócio a fim de armazenar informações específicas do usuário.

2.2.3 UWE [KOCH et al, 2001]

A abordagem UML-based Web Engineering (UWE) apresentada por [KOCH, 2001], dá suporte ao desenvolvimento de aplicações Web com foco especial na sistematização e personalização. Trata-se de uma abordagem orientada a objetos com base no princípio de duas dimensões: tempo e conteúdo, do Processo Unificado [JACOBSON, BOOCH E RUMBAUGH, 1999]. O processo que abrange a totalidade do ciclo de vida de aplicações Web é definido com base em fases, fluxos de trabalho e metas. As fases são: criação, elaboração, construção de transição e manutenção.

Os principais aspectos da abordagem UWE são: (i) o uso de uma notação padrão, neste caso, UML; (ii) a definição precisa do método, ou seja, a descrição detalhada das diretrizes e (iii) a especificação de restrições, para a precisão dos modelos.

Na modelagem da hipermídia é dada grande importância ao processo de autoria que consiste em três etapas: análise de requisitos, projeto de navegação e projeto de apresentação. No fim do processo, os seguintes artefatos são produzidos:

- Modelo de caso de uso;
- Modelo conceitual;
- Modelo de navegação do espaço e modelo de estrutura de navegação;
- Modelo de apresentação;

Na modelagem dos casos de uso, é feita a análise de requisitos cujo principal objetivo é encontrar os necessidades funcionais da aplicação web e representa-las como casos de uso.

O objetivo do projeto conceitual é a construção de um modelo do domínio da aplicação levando em conta os requisitos capturados com casos de uso.

Técnicas tradicionais orientadas a objetos são usados para construir o modelo conceitual, tais como associação de classes e definição de estruturas de herança.

Com base no modelo conceitual, o método de navegação propõe um conjunto de diretrizes para a construção de um modelo que represente o espaço e a estrutura de navegação. O método inclui um conjunto de elementos estereotipados de modelagem UML para o projeto, como índices, visitas guiadas, consultas e menus. Esses estereótipos são usados na construção de diagramas de classe UML para representar o modelo de espaço de navegação e o modelo de estrutura de navegação.

Modelagem de apresentação visa o *design* de interfaces abstratas com o usuário e o *design* da interação do usuário com o aplicativo Web, e é feita em dois passos: O primeiro passo define uma visão da interface com um esboço do conteúdo e o "*look and feel*" das páginas. Essas visões da interface podem então ser combinadas para a montagem de outros cenários. A segunda etapa se concentra na dinâmica da apresentação representada com diagramas de seqüência UML.

2.2.4 OOWS [PASTOR et al, 2003]

OOWS (*Object Oriented Web Solution*) é uma abordagem metodológica para desenvolver aplicações web em um ambiente de desenvolvimento orientado a objeto. Esse ambiente de trabalho integra modelos adequados para capturar a estrutura, comportamento, navegação e requisitos de apresentação de uma aplicação web.

De acordo com a abordagem OOWS existem duas etapas principais: modelagem conceitual e desenvolvimento da solução. Na fase de modelagem conceitual é obtida a especificação do sistema fazendo uso de modelos.Esta fase é dividida em três sub-etapas:

- **Levantamento de requisitos funcionais:** São técnicas baseadas em casos de uso e cenários e são aplicadas para construir um esquema conceitual.
- **Modelagem conceitual clássica:** Usando modelos estruturais, dinâmicos e funcionais, a estrutura e comportamento do sistema são capturados.
- **Modelagem da apresentação e da navegação:** Um projeto de navegação é construído para modelar requerimentos navegacionais a partir do diagrama de classes. Uma vez construído o modelo de navegação, os requisitos de apresentação são especificados usando um modelo de apresentação que é fortemente baseado no modelo de navegação.

A terceira sub-etapa da modelagem conceitual é a mais importante. Nela são capturados os requisitos de navegação da aplicação web por meio da definição de uma "visão de navegação" (mapa de navegação) para cada tipo de usuários relevantes do sistema. Esta visão fornece uma estrutura de acesso (que define todos os caminhos possíveis de navegação) e personaliza o sistema, dependendo do tipo de usuário.

Uma vez que o modelo de navegação é construído, se faz necessário especificar os requisitos de apresentação das aplicações web. Para este fim, o modelo de apresentação é introduzido. Esse modelo usa os elementos de navegação como base para definir as propriedades de apresentação. Requisitos de apresentação são especificados por meio de padrões simples, que devem estar associados aos elementos do contexto de navegação.

A ultima etapa no processo de modelagem de aplicações web usando OOWS, e a implementação. A partir do mapa de navegação, é possível a obtenção sistemática do esqueleto da aplicação web, que é composto por um conjunto de páginas interligadas. Essas páginas representam a interface (visão estruturada do sistema) e fornecem o acesso às informações e funcionalidade para os usuários.

2.2.5 OO-H [GOMES & CACHERO, 2003]

O método OO-H (Object-Oriented Hypermedia) segue um modelo genérico, baseado no paradigma orientado a objetos, que fornece ao *design*er, com semântica e notação necessária, o desenvolvimento de interfaces com usuário na web e sua conexão com módulos de aplicação lógica pré-existentes. OO-H define um conjunto de diagramas, técnicas e ferramentas que constituem uma boa abordagem para a modelagem de interfaces web. A proposta inclui:

- Processo de *design*;
- Catálogo de padrões;
- Acesso a diagrama de navegação;
- Dupla camada de apresentação (diagrama de apresentação abstrata e diagrama de *layout* composto);
- Ferramenta CASE que suporta e automatiza até certo ponto o processo de desenvolvimento.

O método *Object-Oriented Hypermedia* é centrado pelo conceito de autoria na fase de desenvolvimento do produto. OO-H segue uma “filosofia de integração” e estende abordagens tradicionais de engenharia de software (baseado em UML) com dois modelos: Modelo de Navegação e Modelo de Apresentação.

O modelo de navegação é construído a partir de uma compilação dos casos de uso da UML com a perspectiva centrada no usuário, o que é de grande importância em ambientes web. Ainda faz-se uso dos diagramas de classe, como mecanismo adequado, não só para mostrar o domínio do modelo conceitual como também para expor as interfaces que permitem a conexão com os módulos lógicos pré-existentes na camada de negócios.

Partindo do modelo de navegação, um modelo padrão de apresentação abstrata pode ser obtido de forma automática. Este modelo de apresentação abstrata é composta por um conjunto de templates XML que reúne projeções ortogonais de OO-H, que presisam ser levados em consideração em se tratando de interface web, e que são: (i) a estrutura de interação do usuário, (ii)

navegação, (iii) a lógica do cliente, (iii) composição de página, (iv) *layout*, (v) referências externas e (vi) detalhes da conexão com a camada de negócios.

2.2.6 Compilação das Modelagens Apresentadas

Embora utilizem diferentes formalismos e notações, todas as metodologias citadas se assemelham em (i) o projeto conceitual do domínio da aplicação, (ii) o projeto navegacional definindo a estrutura da apresentação da hipermídia, (iii) o projeto de apresentação especificando a renderização de objetos de navegação, (iv) definição de diferentes visões sobre o ambiente web e (v) o processo centrado no usuário [FIALA ET al, 2004].

Algumas das metodologias que apresentamos usam modelos explícitos permitindo compor o *layout* de apresentações a partir de elementos abstratos da interface de usuário (também chamados *abstract data views* em OOHDM ou *user interface views* em UWE). Por outro lado, métodos como WebML não incluem um modelo específico para expressar apresentação no nível conceitual e fazem a apresentação por meio de folhas de estilo XSLT. Utilizar XSLT para definir interface é uma boa alternativa para manter conteúdo e *layout* estritamente separados.

OOWS define um conceito de visões sobre a navegação independente para cada usuário. Isso é importante para o processo de personalização da hipermídia. Esse conceito é caracterizado pela criação de mapas de navegação. Para um sistema multi-usuário, é importante haver mecanismos de interface que delimitem as ações dos usuários sobre o ambiente. Além disso, sugerem-se mecanismos que disponibilizem elementos de apresentação específicos para cada usuário.

O método OO-H oferece o suporte de ferramenta CASE para automatizar o processo de desenvolvimento. Esse aparato facilita a ação de autoria, que é uma das características em OO-H, útil para aprimorar a usabilidade e acessibilidade do usuário sobre o ambiente.

Para definir a nossa proposta, buscamos realizar uma análise nas metodologias apresentadas e selecionar as características mais pertinentes ao projeto MOrFEu no que diz respeito a apresentação dos espaços virtuais, incluindo a descrição do modelo de navegação abstrata abordado em OOHDM, a importância dada sobre manutenção da hipermídia, como em uma das etapas de UWE, a utilização de XSLT como instrumento de separação de conteúdo e *layout* (WebML), o conceito de visões sobre a navegação da hipermídia, visto em OOWS, e a valorização do processo de autoria (OOH).

2.3 DIVISÃO ENTRE ESTRUTURA, *LAYOUT* E CONTEÚDO

No projeto MOrFEu, o oferecimento do “flexível” em todo ambiente, é a característica que deve sobrepujar na construção dos espaços virtuais, e isso também é fato, no que diz respeito à criação dos templates.

De acordo com [KERER & KIRDA, 2001] a flexibilidade completa de *layout* só pode ser alcançada se houver uma estrita separação entre a estrutura com conteúdo e o *layout*. No entanto, na maioria dos sites, as informações de conteúdo e aparência estão fortemente acopladas em arquivos HTML. Então, para modificar o *layout*, primeiro as informações de conteúdo têm que ser identificadas. Depois, elas precisam ser extraídas. O último passo é integrar as informações do conteúdo extraídas com o novo *layout*. Todo o processo pode tornar-se complicado e difícil, especialmente se o seu formato ou o conteúdo não estiver bem estruturado.

Percebemos que se mantivermos o espaço virtual, independente de sua aparência, e independente das produções dos usuários, a manutenção de todo o ambiente seria muito mais fácil de ser realizada. Levando em conta o fato de que a todo o momento os participantes do ambiente poderão editar suas produções, a estrutura do ambiente e sua interface, é necessário proporcionar ferramentas independentes para cada uma dessas camadas.

Para resolver o problema de separação entre o espaço virtual e sua apresentação, existem tecnologias como XML e XSL que proporcionam

completa independência entre esses elementos, e no momento, formam o conjunto de ferramentas mais adequado para trabalhar na solução da separação entre estrutura e *layout* [KERER & KIRDA, 2001].

Concluímos que para o que o editor atuante sobre a camada de apresentação do MOrFEu funcione da melhor maneira possível, temos que manter uma estrita separação entre a aparência do ambiente, o ambiente em si e o conteúdo (produções dos usuários). Uma vez isso alcançado, é garantido que o editor funcionará sem interferência de nenhum elemento da hipermídia que não seja um elemento de interface.

2.4 USABILIDADE E PADRÕES DE INTERAÇÃO

Pela definição dada em [ISO9241, 1993], a usabilidade é "a capacidade que um sistema interativo oferece a seu usuário, em um determinado contexto de operação, para a realização de tarefas de maneira eficaz, eficiente e agradável".

A usabilidade tornou-se uma disciplina e uma área da engenharia de software preocupada com o desenvolvimento de sistemas ao mesmo tempo úteis, eficientes e agradáveis. Os estudos de usabilidade, de uma maneira geral, se propõem a viabilizar o desenvolvimento e/ou adequação de sistemas de informação a partir de uma perspectiva centrada no usuário, em suas necessidades, tarefas a serem executadas, condições de interação com o sistema, entre outros fatores.

A meta principal no desenvolvimento de interfaces é aproximar o modelo mental do usuário do modelo do programa, com a proposta de elaborar um projeto centrado no usuário. Nessa proposta, o usuário não tem que se adaptar ao programa, mas o programa ao usuário, a fim de que a usabilidade seja fator preponderante no projeto.

2.4.1 O Usuário como Centro do Projeto

Em um projeto de interface de usuário, devemos entender que, para a maioria dos usuários, a interface é na realidade, o próprio programa. Pouco interessa a ele em qual linguagem o programa foi desenvolvido, ou se é cliente-servidor, monousuário; o que ele quer, é que o programa esteja ali pronto para suprir suas necessidades [VALENTE, 2004].

De acordo com [WELIE, 2000], quando se toma a perspectiva do usuário, torna-se importante pôr ênfase na argumentação de “por que” e “como” a usabilidade é melhorada. Sem isso é impossível dizer quando e como a solução é boa ou apenas aceitável. Uma vez que o centro do projeto é o próprio usuário, é a ele que essa argumentação deve ser dirigida.

De acordo com [VALENTE, 2004], ao centrarmos o projeto no usuário final, concedemos a ele poder que este não tinha antes: de influir qualitativamente no processo de projetar interfaces de usuário, como um amador projetando um jardim ou a melhor maneira de se colocar uma cerca. Com a colaboração ativa dos usuários no processo, ganhamos um importante aliado para a melhoria da qualidade do produto final.

2.4.2 Interação Humano-Computador

A interação humano-computador - IHC, é uma área multidisciplinar, envolvendo disciplinas como Ciência da Computação, Psicologia, Ergonomia e Sociologia para tentar abranger o conceito de um ser humano interagindo com um computador. Em uma definição mais formal, é a disciplina preocupada com o projeto, avaliação e implementação de sistemas computacionais interativos para o uso humano e com o estudo dos principais fenômenos ao redor deles [HEWETT, 1992]

Um dos objetivos da interface com o usuário é alcançar alto grau de qualidade de interação entre o usuário e o ambiente. Para a produção de interações produtivas e com qualidade, é necessário o desenvolvimento do projeto de interação humano-computador.

Segundo [VALENTE, 2004], o projeto de interação é um processo onde o objetivo é desenvolver um espaço de trabalho através de uma interface, na qual o modelo mental do usuário alvo seja correspondido pelo modelo do programa. A interface gerada a partir deste projeto é basicamente uma resposta às necessidades do usuário dentro do contexto abordado. Um bom processo de projeto de interação deve ter fundamentalmente três características:

- Deve ser focalizado no usuário, desde o início do projeto até a avaliação do artefato final;
- Deve identificar, documentar e definir metas de usabilidade relativas a experiências prévias de usuários específicos;
- Deve ser iterativo, a fim de produzir uma melhoria contínua.

Existem alguns artefatos, aplicados nos projetos de interação, que delimitam a forma como os usuários interagem com sistemas computacionais. São os estilos de interação.

Um estilo de interação bastante conhecido é a **Manipulação Direta (MD)**, que permite ao usuário agir diretamente sobre os elementos representados na tela. Na MD, as metáforas são fortemente utilizadas, onde o cursor representa uma extensão da mão do operador e os objetos exibidos são considerados "reais" no espaço do sistema. O usuário clica, arrasta, solta etc. É o principal estilo utilizado na realidade virtual.

2.5 MANIPULAÇÃO DIRETA E INTERFACES EXPLORÁVEIS

Segundo [ORLANDO, 2006] o termo manipulação direta foi criado em 1982 por Ben Shneiderman para definir um modo de interação presente em sistemas e produtos como os videogames. Ele retoma com frequência em suas publicações as definições, características e vantagens das interfaces baseadas neste conceito, defendendo sua ampla adoção.

[TURKLE, 1997] menciona a importância das interfaces exploráveis com relação ao aumento da capacidade de interação dos utilizadores com os sistemas computacionais que valorizam a manipulação direta. Nas palavras de Turkle fica evidenciada a influência da exploração informal das interfaces:

> "Estas novas interfaces projetam uma mensagem do tipo: "Brinca comigo, faz experiências, há mais de um caminho possível". A nova estética de concepção de software afirma efetivamente que os utilizadores não devem ser obrigados a trabalhar com sintaxe; deve-lhes ser permitido brincar com imagens, formas, cores e sons. Os utilizadores de computadores não deverão ter que se preocupar com a complexidade de uma linguagem de programação; devem ser-lhes fornecidos objetos virtuais que possam ser manipulados tão diretamente quanto possível. [...]" [TURKLE, 1997]

A manipulação direta é normalmente considerada como a forma de se instituir sistemas exploráveis, ou melhor, ambientes em que se possa aprender a usar por exploração das funcionalidades e princípios de operação, além de produtores de menor ansiedade e rejeição por parte dos iniciantes. A Técnica de MD muda a maneira de interagir e de pensar sobre computadores. Shneiderman define os princípios do conceito de MD como:

- Representação contínua de objetos e ações de interesse;
- Ações físicas e movimentos (com mouse, joystick, telas sensíveis a toque etc.) e "botões" rotulados na tela para serem clicados, ao invés de comandos por sintaxe complexas;
- Operações incrementáveis, rapidamente reversíveis, cujos efeitos sobre os objetos de interesse são imediatamente visíveis.

A aplicação dos princípios do conceito de manipulação direta, explica [SHNEIDERMAN, 1997], permite dar aos sistemas alguns atributos considerados importantes, benéficos e desejáveis:

- Os participantes do ambiente não se preocupam em errar uma vez que suas ações podem ser revertidas com facilidade;

- Eles ganham confiança, pois são os iniciadores das ações e sentem-se no controle do ambiente;
- Iniciantes podem aprender de modo rápido as funcionalidades básicas, uma vez que eles são os exploradores;
- Experts podem trabalhar com rapidez realizando uma ampla gama de tarefas e mesmo definindo novas funções e recursos;
- Usuários podem ver imediatamente se suas ações estão se encaminhando para seus objetivos e, se não, simplesmente mudar a direção de sua atividade;

Segundo [ORLANDO, 2006], dois elementos caracterizam os maiores objetivos da manipulação direta como eixo do *design* de interfaces: dotar o usuário de poder para executar suas ações e ao mesmo tempo tornar a experiência de uso prazerosa, deixando-o satisfeito. Para alcançar os propósitos do projeto de interfaces adequadas, é necessário que se cumpram as metas esboçadas como importantes pelo discurso do *design*, geralmente associadas ao conceito de usabilidade: produzir interfaces fáceis de usar, de aprender, de memorizar, que permitem uma experiência agradável e produtiva, na qual ansiedade e erros sejam diminuídos.

Vale ainda destacar o argumento da satisfação subjetiva [SHNEIDERMAN, 1997]: usuários entusiasmados com sistemas baseados em manipulação direta costumam relatar sentimentos de domínio da interface, de competência na execução de tarefas, facilidade de aprender o sistema de início e de assimilar recursos avançados, confiança na capacidade de reter mestria, prazer em usar o sistema, avidez em mostrá-lo para iniciantes e desejo de explorar aspectos mais poderosos dele.

2.5.1 Modelo WYSIWYG

WYSIWYG é o acrônimo da expressão em inglês "**W**hat **Y**ou **S**ee **I**s **W**hat **Y**ou **G**et", cuja tradução remete a "O que você vê é o que você obtém". O modelo está ligado à capacidade de um programa de computador de permitir que um documento manipulado na tela, tenha a mesma aparência de sua forma final,

impressa ou como interface entre um objeto do mundo real e outro do mundo virtual. O uso inicial do termo estava relacionado a editores de texto, contudo, atualmente, tem sido aplicado a qualquer tipo de programa.

Em se tratando de web, a maioria dos aplicativos WYSIWYG lançados atualmente são editores de páginas da internet, que procuram facilitar o trabalho do usuário, que podem criar ou editar páginas na internet sem precisar conhecer linguagens de programação, como no exemplo da criação de um *post* em um *Blog* na Figura 2.1.

Figura 2.1: WYSIWYG na web. Exemplo de um *Blog*

De acordo com [ORLANDO, 2006], o modelo WYSIWYG, em que se cria uma relação entre o que aparece na tela e o que seria o resultado obtido pelo usuário, é a base da manipulação direta, cuja origem remontaria ao momento em que a tela passou a ser pensada como folha de papel. Esta possibilidade de uma “representação” ou de uma simulação que guarde forte proximidade visual é então, fundamental.

2.5.2 Manipulação Direta e Aprendizagem

Shneiderman apresenta um fundamento para advogar que a representação visual de problemas a serem resolvidos, de conceitos matemáticos e de outras formas mais abstratas de pensamento, recebem um tratamento adequado e eficiente quando concretizados por objetos e modelos que permitam manipular estes conceitos, como é o caso do ábaco em relação a números e operações matemáticas.

Lembrando de Jean Piaget, Shneiderman argumenta que a manipulação direta e sua ligação com o movimento corporal, com a ação sobre objetos, está mais vinculada às fases de desenvolvimento do operacional e concreto, do que ao pensamento abstrato e à manipulação de símbolos, favorecendo as condições de aprendizagem das interfaces [ORLANDO, 2006].

Nesse contexto, nos deparamos com o conceito de "*information visualization*", que caracteriza o uso dos computadores para suportar representações visuais e interativas de dados abstratos de modo a amplificar a cognição. O pressuposto é o papel da capacidade humana de processamento visual da informação: o uso de princípios de *design* como a manipulação direta e o controle dinâmico pelo usuário ajuda a enfrentar as dificuldades de dominar o ambiente, facilitando a compreensão e reduzindo a ansiedade, com resultados consistentes tanto para usuários experientes como iniciantes [AHLBERG & SHNEIDERMAN, 1999].

2.5.3 Importância de Interfaces Flexíveis em Ambientes Virtuais de Aprendizagem

A flexibilidade e a eficiência de uso estão ligadas à capacidade do ambiente em se adaptar ao contexto, às demandas e às preferências do público, a fim de incrementar sua eficiência. No caso de ambientes virtuais de aprendizagem que são utilizados por uma diversidade de usuários, é indispensável manter interface flexível de modo que seja possível realizar uma determinada tarefa de

diferentes maneiras, em consonância com a realidade do contexto e com as singularidades dos indivíduos [GUEDES, 2009].

As interfaces, em se tratando de AVAs, vêm para facilitar a operação sobre o objeto no qual os participantes trabalham. Nesse caso, é necessário colocar à disposição dos usuários, meios que lhe permitam personalizar a apresentação de seus espaços virtuais a fim de levar em conta as exigências da tarefa, de suas estratégias ou de seus hábitos de trabalho.

Um dos principais aspectos que cercam a MD é justamente flexibilidade na apresentação do sistema, que deve ser vista como uma competência da interface em se adaptar às variadas ações do público, permitindo a chance de execução de uma mesma tarefa de diferentes maneiras, recorrendo a diferentes procedimentos, os quais podem ser mínimos ou não.

Flexibilidade em interface somada ao conceito de consideração da experiência individual, resumem uma nova definição: adaptabilidade. Adaptação refere-se à capacidade de reagir conforme o contexto e em consonância com as necessidades e preferências do usuário.

2.6 TRABALHOS CORRELATOS

Encontramos na comunidade acadêmica alguns trabalhos que têm como foco interface com usuário. Especificamente, buscamos projetos que tratem o objeto “apresentação” como moldável, e possivelmente adaptável às necessidades ou características de seus utilizadores.

Nesta seção serão abordadas algumas tecnologias que atuam de uma forma similar ao Editor de Templates, com relação às características da arquitetura, a utilização de conceitos das interfaces exploráveis, segregação entre estrutura e aparência e utilização de templates customizáveis.

2.6.1 Gerador de Interface [PETERMANN, BELLIN & KROTH, 2001]

O Gerador de Interface (G.I) utiliza como base a integração de funcionalidades oriundas do DHTML (JavaScript e Cascading Styles Sheets) e também Java Swing. O G.I define como templates as propriedades essenciais de cada componente de interface, incluindo a definição de características da aparência de uma interface, tais como cor e tamanho de fonte de botões, *labels* etc.

Todas as informações que caracterizam a interface são armazenadas na base de dados para posterior utilização na geração de formulários. Cada formulário apresenta um *layout* que tem como finalidade a construção de um padrão para a disposição dos vários elementos que compõem uma interface.

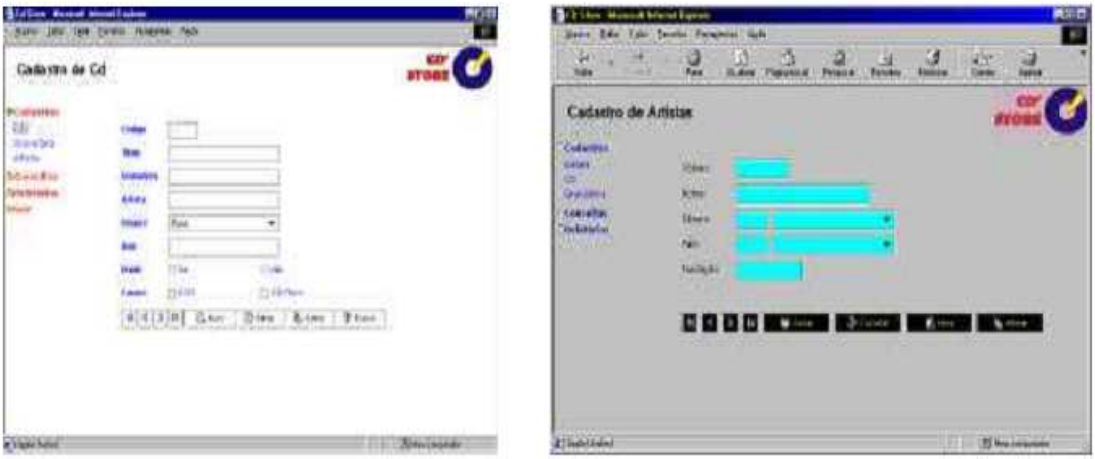


Figura 2.2: Exemplos de interfaces geradas a partir do G.I

Os templates no G.I provêm a identidade da estrutura dos *sites* produzidos. Como se observa na Figura 2.2, dois modelos de interfaces que, apesar de terem a mesma funcionalidade (formulário), são apresentados de maneiras diferentes para usuários diferentes.

O G.I se assemelha ao editor de templates ao ser definido sob conceitos de interfaces exploráveis. Contudo, ele foi desenvolvido para se portar como um programa desktop que gera HTML para formulários web, e isso acarreta problemas de portabilidade que não existe no editor de templates, uma vez que é uma tecnologia baseada em web.

2.6.2 **HUMANOID [SZEKELY, LUO, & NECHES, 1993]**

Trata-se de um sistema baseado em modelo. Interfaces são especificadas através da construção de um modelo declarativo de como a interface deve parecer e se comportar. Um módulo de suporte padrão que funciona em tempo de execução realiza o modelo para construir o *display* da aplicação e interpretar as entradas de acordo com as informações no modelo.

O HUMANOID apresenta uma interface para a descrição do *design* de páginas que inclui funcionalidades para edição de rótulos, aparência e *layout* do modelo. Além disso, também permite a geração de novos modelos a partir de outros pré-existentes.

O problema dessa tecnologia está em definir interface por meio de modelos declarativos. Assim, o usuário não tem a ideia final de sua interface enquanto a constrói, ao contrário do que seria caso o modelo fosse baseado no paradigma WYSIWYG.

2.6.3 MyXML [KIRDA & KERER, 2000,2001]

A ferramenta MyXML provê geração automática de templates para o website do Vienna International Festival (VIF) e que pode ser aplicada em diversas situações nas quais se busque uma solução flexível e rápida para geração de interfaces web.

Trata-se de um sistema totalmente baseado em tecnologias XML/XSL, sendo o conteúdo e estrutura representados em documentos XML bem estruturados. Toda informação de *layout* é adicionada em arquivos XSL separados, o que proporciona total separação entre *layout* e conteúdo.

O processo MyXML, representado na Figura 2.3, começa com um documento bem estruturado como entrada. Posteriormente, um pré-processamento aplica uma folha de estilo para adicionar informações de *layout* e/ou conteúdo estático. Na próxima etapa, o arquivo intermediário gerado é analisado pelo *MyXML Template Engine* e as tags do namespace (XML) MyXML são

resolvidas. Uma outra transformação XSL (*post-processing*) pode ser aplicada no arquivo de saída para gerar o documento final (geralmente HTML).

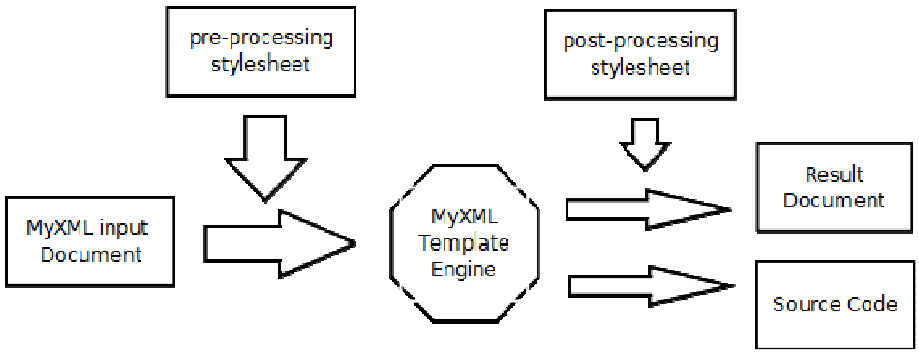
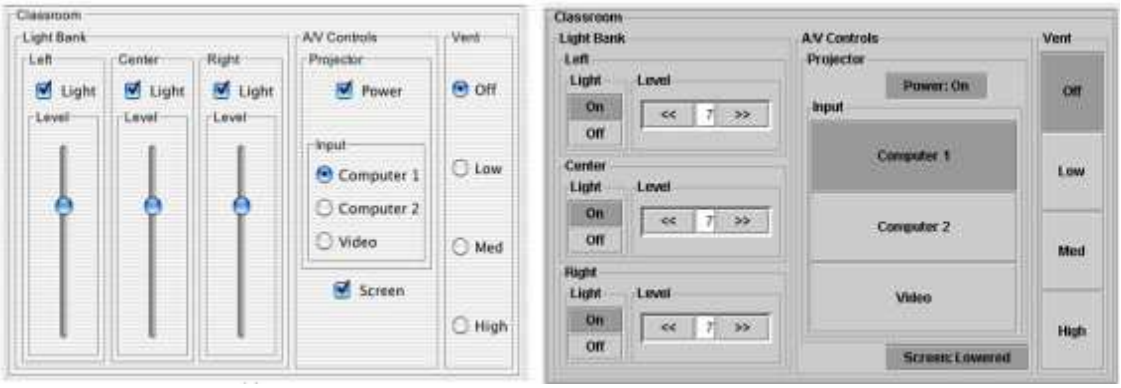


Figura 2.3: Processo MyXML

É uma iniciativa interessante, e com a ideia de separação de conteúdo e *layout* aproveitada no Editor de Templates, porém não apresenta uma solução gráfica para as operações dos indivíduos, ou seja, tudo é feito com codificação o que não a torna atrativa ao usuário final.

2.6.4 SUPPLE [KRZYSZTOF & WELD, 2004]

No projeto SUPPLE existe a premissa que a renderização de uma interface deve refletir as necessidades e padrão de uso de usuários individualmente. Dada a ampla gama de tipos de dispositivos, métodos de entrada, necessidades pessoais e estilos de interação, é inviável para os programadores humanos criar interfaces para cada tipo de dispositivo e todo tipo de usuário. Em vez disso, uma solução automatizada é necessária.



(a) Dispositivo “drag’n drop” **(b) Dispositivo baseado em cliques**

Figura 2.4: Interface "classroom" para dois dispositivos com um mesmo objetivo

Quando é solicitado a SUPPLE prestar uma interface (especificada funcionalmente) para dispositivo e usuário específicos, o sistema realiza uma busca por um modelo que atenda as restrições do dispositivo e minimiza o custo estimado (esforço) da atividade da pessoa. Por exemplo, a Figura 2.4 mostra duas interfaces diferentes para dois dispositivos com o mesmo tamanho de tela, mas métodos diferentes de interação para cada usuário.

Nosso Editor de Templates se assemelha ao SUPPLE quando oferece possibilidade de ter diferentes interfaces para uma mesma estrutura. Contudo, a abordagem SUPPLE trata geração de interfaces como um problema de otimização, no qual o algoritmo decide qual interface é mais adequada ao usuário. No Editor de Templates o usuário é quem define a apresentação de seus dispositivos. Dessa forma, não é a máquina que adapta a interface ao usuário, mas o próprio usuário é quem define como a apresentação de seu dispositivo se dará.

2.7 CONCLUSÃO

Neste capítulo levamos em consideração vários aspectos que surgiram no estudo sobre as interfaces e que serviram de base para o levantamento das características do Editor de Templates.

Iniciamos o levantamento com estudos sobre o que se fala atualmente a respeito de modelagem de hipermídias, selecionando dentro das principais abordagens o que é mais proveitoso para o Editor de Templates. Posteriormente introduzimos os estudos sobre padrões de usabilidade e manipulação direta, no qual descrevemos a importância de uma boa interface em projetos de aprendizagem. Finalmente falamos sobre os trabalhos correlatos ao Editor de Templates.

O referencial teórico auxiliou a elaboração dos conceitos construtivistas que cercam esse trabalho, assim como direcionou as decisões de projeto e ajustes sobre a definição do editor para a apresentação dos veículos de comunicação do MOrFEu.

CAPÍTULO 3 MORFEU E OS TEMPLATES

Neste capítulo abordaremos o meta-ambiente MOrFEu. Nele, definiremos os objetivos dessa nova iniciativa, suas camadas e como acontecem as operações entre elas. Também apresentaremos alguns trabalhos desenvolvidos que estão relacionados ao MOrFEu.

Ainda falaremos sobre os templates e seus componentes e discutiremos o processo de criação da apresentação dos ambientes e o papel dos templates nesse processo.

3.1 MORFEU – MULTI-ORGANIZADOR FLEXÍVEL DE ESPAÇOS VIRTUAIS

Segundo [MENEZES, 2008], é necessário repensar a concepção dos espaços virtuais, facilitando sua apropriação para que seja realizada de forma mais rápida e efetiva, e ao mesmo tempo, oferecer objetos mais adequados a essa apropriação.

O caráter de arquiteturas pedagógicas (que na verdade são tecnologias flexíveis, maleáveis e adaptáveis) deve fazer com que a aprendizagem seja um processo de criação de novidades, de descobertas e invenções, permitindo que as pessoas realizem experimentações, simulações em busca de soluções para questões significativas do ponto de vista do indivíduo.

A proposta do Multi-Organizador Flexível de Espaços Virtuais, o MOrFEu, foi concebida a partir da percepção da necessidade de flexibilidade nos ambientes virtuais existentes. O objetivo é moldar os espaços de acordo com os interesses, ideias e, principalmente, necessidades e afetividades dos usuários, sob a premissa de que emoções afetam a aprendizagem [PIAGET, 1989].

Buscamos a produção de ambientes inteligentes para mediação de atividades cooperativas na web, com ênfase no apoio à aprendizagem e à gestão do conhecimento.

O Ambiente MOrFEu apresenta elementos estruturantes que viabilizam a descrição de diversos ambientes, entre eles os conhecidos espaços virtuais da web 2.0. A princípio, qualquer ambiente colaborativo que possa ser descrito de forma hierárquica, pode ser representado. Exemplo desses ambientes são os fóruns de discussão, os *Chat*s, os *Blog*s, os *Wiki*s e ambientes de redes sociais. Nosso objetivo é a descrição de novos espaços virtuais que não podem ser representados nesses ambientes conhecidos ou cujo uso demandaria um acentuado esforço por parte dos usuários.

Em MOrFEu, toda e qualquer produção individual é registrada em uma unidade de produção intelectual (UPI), que é gerenciada e versionada pelo sistema. O espaço de colaboração é descrito por meio de um documento composto, organizado de forma hierárquica, sub-espaços, e populados pelas unidades de produção intelectual por meio de um mecanismo de publicação. Neste sentido, tanto uma postagem quanto um comentário, possuem o mesmo tratamento, diferindo apenas com respeito à permissão e prazo de submissão [MENEZES, 2008].

As produções compostas de UPIs são tratadas por um Veículo de Comunicação (VCom). Cada VCom possui suas diretrizes de composição. Por exemplo, um *Blog* é um VCom no qual as produções são organizadas em forma de pilha, ou seja, o mais recente está no topo.

Os espaços estruturados podem ser apresentados e explorados de forma diferente, seguindo a descrição de um template, que pode ser customizada segundo o seu usuário.

Nas próximas seções discutiremos as quatro principais estruturas que compõem o MOrFEu, e que provêm: (i) controle, (ii) conteúdo, (iii) forma e (iv)

estilo. São elas: o Núcleo, o Editor de UPIs (Unidade de Produção Intelectual), o Editor de VCom (Veículo de Comunicação) e o Editor de Templates.

3.1.1 NÚCLEO

O núcleo do MOrFEu é representado por uma arquitetura de software que envolve vários elementos, camadas e componentes. Uma de suas funções é a delimitação e descrição das operações que os espaços virtuais poderão sofrer, incluindo as composições e as visões que os usuários terão sobre os ambientes produzidos.

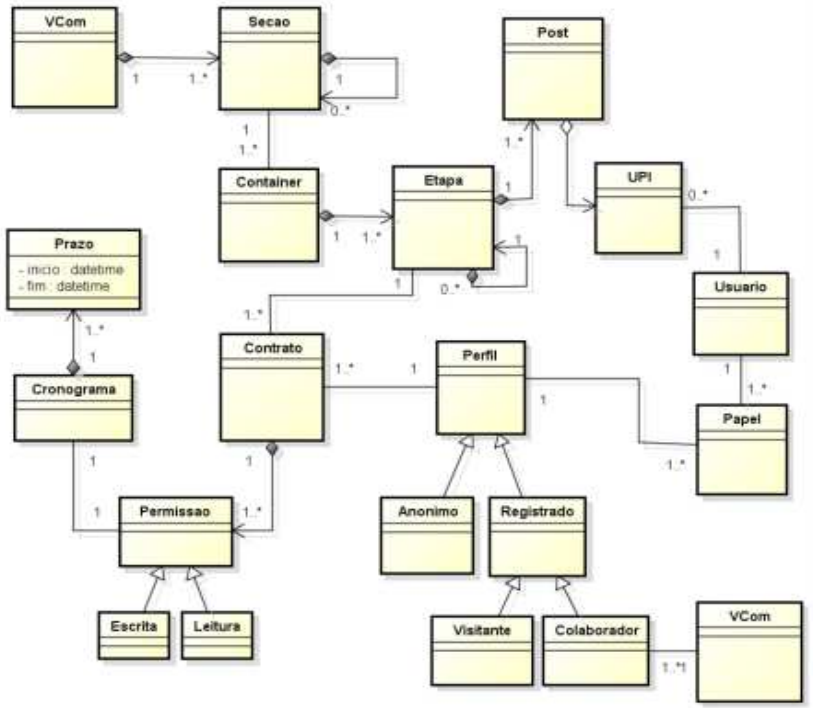


Figura 3.1: Diagrama de classes - Arquitetura do MOrFEu [VIEIRA, 2011]

No trabalho elaborado por [VIEIRA, 2011], encontramos a discussão sobre as estruturas que compõem o núcleo. No diagrama de classes da Figura 3.1, contemplamos algumas dessas estruturas, dentre as quais podemos mencionar o VCom, que reproduz o espaço virtual.

Cada **VCom** possui uma **seção** (ou instância). Em determinados momentos os usuários poderão depositar suas **UPIs**, por meio do **post** (efetuando uma postagem). Os posts poderão existir durante uma **etapa** que deve obedecer às

restrições do **contrato** do espaço virtual, que inclui quem poderá ver/editar UPIs em um prazo pré-determinado.

O núcleo deverá ainda coordenar o versionamento e armazenamento das composições intelectuais dos usuários, que aqui chamamos de Unidade de Produção Intelectual (UPI).

3.1.2 UPI

No MOrFEu, o elemento básico de autoria é a Unidade de Produção Intelectual (UPI), usada para registrar as produções dos usuários. Cada UPI possui um autor, um título e um conteúdo (corpo), e pode ser criada ou editada a qualquer instante pelo usuário.

As produções no MOrFEu são versionadas, ou seja, cada edição de uma UPI resulta em uma nova versão, sendo que cada versão pode ser posteriormente usada e reusada em situações de interação. Por exemplo, uma UPI que foi usada para enviar uma mensagem para um colega por meio de *Chat*, pode também ser usada para publicação em um fórum. [MENEZES, 2008].

O conceito de UPI apresentado permite que o conjunto de produções não fique atrelado às ferramentas. Dessa forma, como se pode observar na Figura 3.2, todo material produzido pelos autores são registrados e versionados e, no momento certo, são mostrados no VCom.

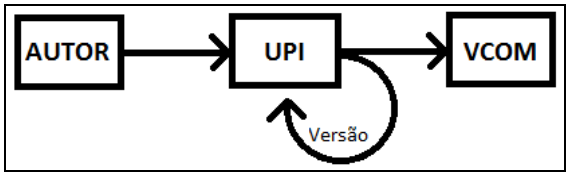


Figura 3.2: Associações da UPI

A ferramenta básica no contexto das UPI é o Editor de UPIs. Esse editor está intimamente ligado ao núcleo, com uma relação de existencialismo, isto é o Editor de UPI só existe se o núcleo existir. A partir dele é permitido ao usuário reiterar seus conceitos, registrar novas ideias, substituir composições

cognitivas que pertençam a algum contexto não mais interessante a ele, dentre outras coisas.

3.1.3 VCOM

Inicialmente, o usuário tem apenas a UPI que desenvolveu e precisa, então, de uma maneira para representá-la. No caso do MOrFEu, os usuários deverão criar seus espaços de trabalho na web, e para isso, eles elaboram VComs. Por definição simples, VCom é a solução para a estrutura que as UPIs deverão seguir em uma página da web, ou seja, estamos falando de um veículo de publicação de uma UPI

Segundo [MENEZES, 2008], um determinado veículo possui características e estruturação próprias, que são passíveis de modificação a qualquer momento. Por exemplo, um Jornal Online, possui perfis de editores e revisores, e se organiza em cadernos. Cada um dos cadernos pode possuir uma ou mais seções temáticas. Algumas dessas seções podem aceitar que seus leitores postem comentários. A postagem é a atividade de publicar a UPI, que está na coleção privada de cada usuário, em uma determinada seção do VCom. Inclusive, as postagens obedecem ao prazo de tempo determinado nas seções.

Com o editor de VComs podem-se elaborar veículos, como os que já estão em uso no meio computacional: *Blog*s, Fóruns, *Wiki*s, *Chat* etc. Porém, a grande vantagem na utilização de VComs está no fato de que eles podem ser modelados para favorecer qualquer tipo de discussão, seja ela síncrona ou assíncrona.

Além dos ambientes colaborativos conhecidos, a cada instante, alguém pode precisar de novas formas de produzir documentos colaborativamente. Para ilustrar, considere a seguinte situação hipotética, que denominaremos de "Confronto de Opiniões".

Imaginemos que algumas pessoas se dividam em dois grupos a fim de discutir sobre um determinado assunto, sendo que um grupo se posiciona a favor e outro contra uma ideia sendo discutida. Os grupos têm um prazo para postar

suas argumentações e apresentar duas provas que a sustentem. Ao final de cinco argumentações e contra-argumentações os grupos deverão chegar a uma conclusão. No final do confronto, o mediador poderá definir um grupo vencedor, por ter exposto melhor suas ideias ou então não declarar um vencedor e apenas expor os erros e acertos de cada grupo.

No momento, não existe ferramenta computacional voltada para esse tipo de ambiente de discussão, apesar da possibilidade de realizar uma mistura de ferramentas da web para elaborar o confronto. A ideia do veículo de comunicação é justamente possibilitar a modelagem de qualquer tipo de espaço virtual sem ter que buscar alternativas marginais.

Cada VCom possui suas diretrizes de composição, que geralmente obedecem os padrões comportamentais das estruturas de dados em programação computacional. Por exemplo, um fórum é um veículo de comunicação no qual as produções estão organizadas em forma de árvore, com a possibilidade de uma UPI “responder” a outra ou iniciar um novo ramo de publicação. Seguindo essa linha, um *Blog* é um VCom no qual as produções são organizadas em forma de pilha, ou seja, o mais recente está no topo. [RANGEL, 2011].

Os VComs permitem a construção de documentos individuais ou através da interatividade dos seus colaboradores. Entende-se que a partir deste processo de cooperação o documento resultante trata-se de uma autoria coletiva, e cada colaborador é um coautor da produção.

A ferramenta básica para a edição/construção de VComs é o Editor de VCom. Na composição do veículo devem ser fornecidas as informações sobre a estrutura do ambiente, isto é, quais elementos o veículo terá (por exemplo, se o VCom for um *Blog*, ele conterá os elementos “titulo”, “postagem” e “comentários”) como e darão as interações entre os participantes e em que momento elas poderão acontecer.

Uma vez definido o VCom, é necessário definir a aparência. Ainda usando o exemplo do *Blog*, cada item da interface (título, postagem e comentários) está

inseridos dentro de “containers”. Na criação da aparência, o usuário deverá escolher o local aonde cada elemento deverá dispor-se na página, determinar as cores de plano de fundo e texto de cada item, escolher como navegar entre os comentários, dentre outras ações.

A mesma flexibilidade que é dada na criação do veículo de comunicação, também deve estar presente na determinação da aparência do VCom, e aí que entram os Templates.

3.2 MODELO CONCEITUAL DE DADOS NO MORFEU

No início da concepção do MOrFEu, foram identificadas as classes, atributos, associações e operações dos elementos que compõem o ambiente. Com posse dessas informações foi possível chegar a um modelo de entidades e relacionamentos da estrutura inicial conforme ilustra a Figura 3.3.

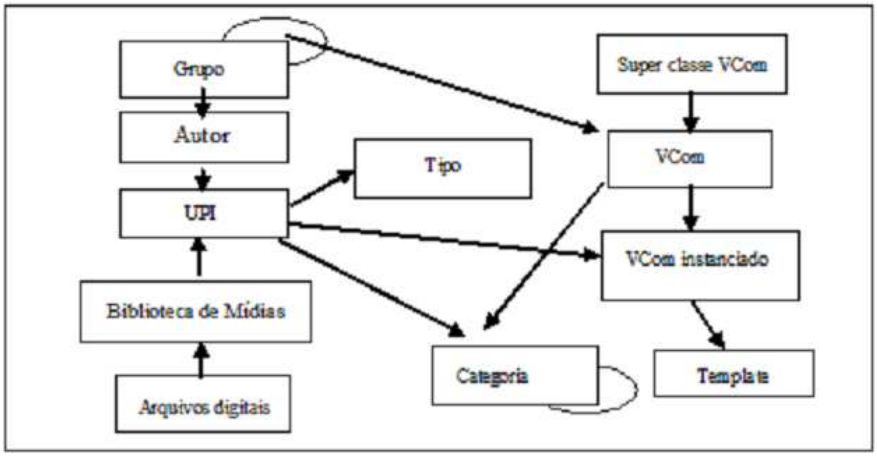


Figura 3.3: Modelo conceitual inicial do MOrFEu [RANGEL et al., 2009]

Tudo começa com a UPI, que pode ser de autoria coletiva ou individual, e deve ser depositada e visualizada em um VCom instanciado. As UPIs podem ser categorizadas ou pertencer a um tipo específico, que seja suportado pela biblioteca de mídias. Os tipos incluem: texto sem formação, códigos fontes de programas, partituras musicais, áudio, vídeo dentre outros.

No MOrFEu conceitual existe uma biblioteca de VComs que oferece mecanismos para que os VComs sejam instanciados a qualquer momento por um usuário que queira ter posse de um novo espaço virtual. O VCom

instanciado, então deve receber o template, que pode ser uma instância de outro template, ou criado a partir o início.

Com relação aos usuários, o MOrFEu apresenta a possibilidade da criação de grupos, no qual os usuários podem ser inseridos afim da delimitação de perfis de autores que sejam ativos ou passivos com relação à escrita/leitura de UPIs.

3.3 PROJETOS RELACIONADOS AO MORFEU

O projeto MOrFEu vem sendo desenvolvido por alguns grupos distribuídos dentro do âmbito nacional, especialmente nas universidades federais do Amazonas (UFAM) e Espírito Santo (UFES) e já apresenta vários pesquisas focadas na sua proposta. Mencionaremos a seguir, alguns trabalhos desenvolvidos recentemente no âmbito da frente de pesquisa da UFES.

Em [VIEIRA, 2011] contemplamos uma proposta de arquitetura, baseada no modelo 3C de colaboração, que é caracterizada pela divisão em camadas, trata-se do Núcleo do MOrFEu. O modelo concebido na proposta tem por objetivo separar os elementos da arquitetura de software em favor da flexibilidade, tanto no projeto, quanto na codificação de um software.

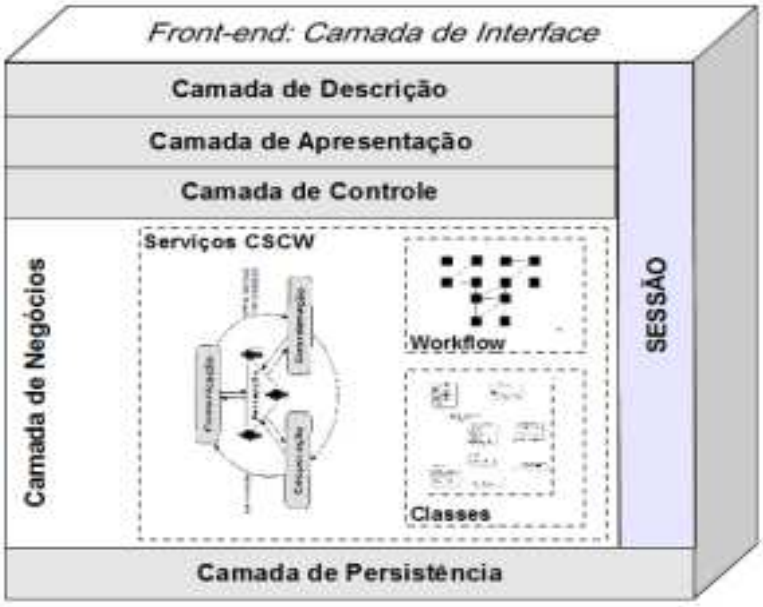


Figura 3.4: Visão geral da arquitetura em camadas do núcleo do MOrFEu [VIEIRA, 2011]

A Figura 3.4 denota a distribuição das camadas. Cada camada interage apenas com a próxima seguindo o fluxo de cima para baixo, enquanto o fluxo contrário estabelece um retorno de informações que foi solicitado, sendo que uma camada poderá requisitar à mais próxima, informações necessárias para seu processamento.

Em outro trabalho, [RANGEL, 2011] apresenta um dos conceitos centrais do MOrFEu: Veículo de Comunicação (VCOM). O VCom baseia-se numa estrutura conceitual que sustenta a construção de ambientes para apoiar as atividades cooperativas com suporte à flexibilidade desejada. Em linhas gerais, o VCom possui suas diretrizes de composição com características e estruturação próprias, que são passíveis de modificação a qualquer momento.

Figura 3.5: Tela de Criação de VCom [Rangel, 2011]

Veículo de comunicação, também pode ser definido como um espaço virtual moldável e flexível que permita a colaboração de seus participantes. A ferramenta principal para a criação de VCom, é o Editor de VComs. A Figura 3.5 mostra a tela de criação/edição de veículos, proposta por [RANGEL, 2011]. Cada veículo criado deverá receber um template que será responsável pela apresentação do veículo.

Citamos ainda o trabalho de [PERUCH & MENEZES, 2010] que apresenta um estudo de caso no qual, atividades colaborativas são descritas utilizando uma ontologia do domínio de colaboração CONTO [FRECHIANI, 2009]. Além disso, nesse trabalho também é realizada uma modelagem do ambiente MOrFEu (Figura 3.6) para que tenha um entendimento comum do domínio, dado que há uma padronização de entidades e do vocabulário utilizado.

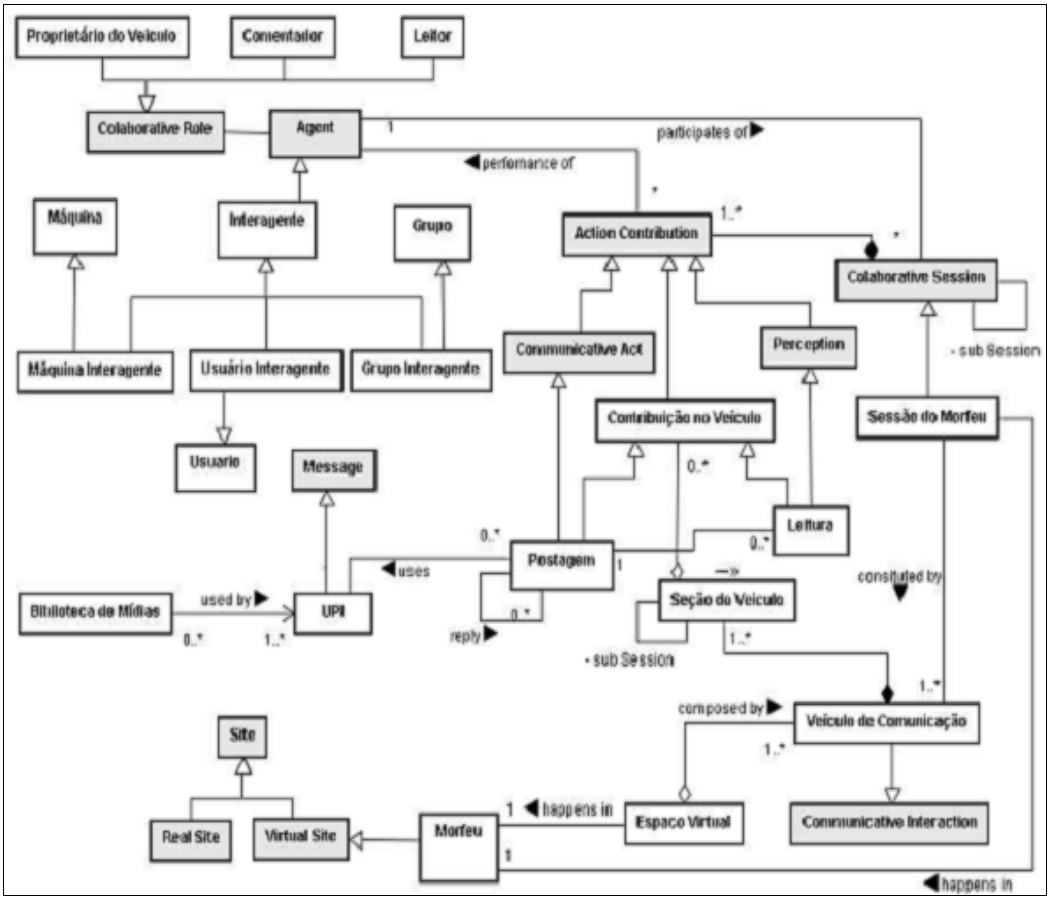


Figura 3.6: Fragmento da ontologia proposta por PERUCH & MENEZES [VIEIRA, 2011]

Outro trabalho influenciado pela concepção do MOrFEu, é o descrito em [NASCIMENTO, 2011], que apresenta algumas diretrizes de acessibilidade para o MOrFEu. Esse trabalho está diretamente ligado aos templates, uma vez que os padrões de acessibilidade são influenciados pelas definições da aparência do ambiente.

Na apresentação das diretrizes sobre acessibilidade, é destacada a importância sobre os templates trabalharem em conjunto com ferramentas ou tecnologias que processem ou interpretem o conteúdo de forma a facilitar o acesso dos usuários ao ambiente. A responsabilidade dos templates fica por

conta de preparar o conteúdo para ser corretamente utilizado por essas tecnologias.

Os templates oferecem boas opções para tratar o aspecto visual dos VComs. Entretanto, quando se pensa em acessibilidade, é necessário considerar as possíveis limitações que os usuários podem apresentar, como insuficiência ocular, daltonismo, dislexia etc.

Por fim, mencionaremos o trabalho desenvolvido por [NATALI, 2011]. Trata-se de um *FrameWork* cujo objetivo é promover artefatos facilitadores para o desenvolvimento de ambientes colaborativos, segundo a abordagem do Modelo 3C de colaboração. Trata-se do *FrameWork* de Ambientes Colaborativos (FrameColab).

O FrameColab possui dois propósitos básicos: fornecer mecanismos para criação (instanciação) de Vcoms e prover o ambiente necessário para usuários consumirem Vcoms instanciados.

Para a criação de novos Ambientes Colaborativos o *FrameWork* proposto deve fornecer mecanismos importantes aos desenvolvedores: (i) possibilidade de descrever estruturalmente Vcoms através de uma linguagem baseada em XML; (ii) possibilidade de acoplar implementações em pontos específicos (*hot-spots*); (iii) definição de um padrão arquitetural interno para o desenvolvimento de aplicações colaborativas; e (iv) acoplamento dos principais conceitos presentes no projeto MOrFEu.

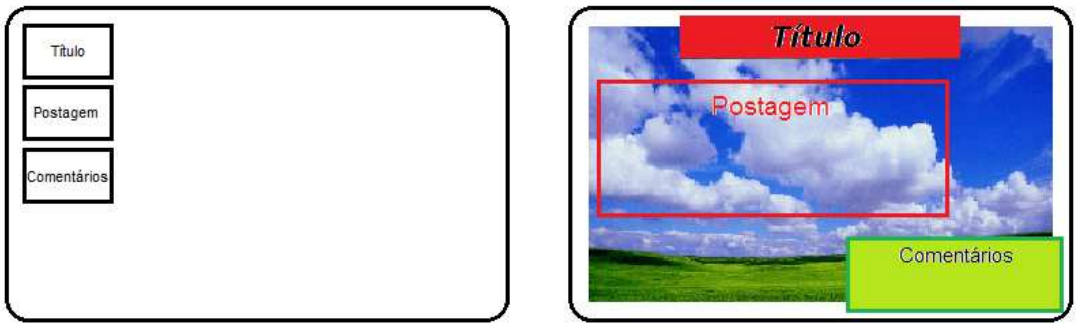
3.4 TEMPLATES COMO FORMA DE APRESENTAR ESPAÇOS VIRTUAIS NO MORFEU

No contexto do MOrFEu, os meios de comunicação, independente do tempo (síncrono ou assíncrono) resultam em documentos, continuamente construídos ao sabor e às necessidades de uma determinada comunidade. Para visualizá-los, concebeu-se o conceito de modelos que podem ser personalizados para as necessidades e gostos dos seus leitores: os templates.

Um Template é, por definição dentro do MOrFEu, a camada de um VCom responsável pela apresentação e pelas visões do sistema. Cada VCom poderá ser visualizado de diferentes formas por seus usuários: (i) sintonizados de acordo com suas preferências, (ii) dependendo da necessidade (pessoas com limitações de percepção) e/ou (iii) disponibilidade de recursos visuais e auditivos (utilizando diferentes equipamentos).

Ao longo do tempo, o ambiente terá disponível, grandes famílias de templates que permitirão visualizações inteiramente particulares para mesmas UPIs. Além disso, o template deve fornecer mecanismos que leiam as restrições do VCom com relação aos usuários no que tange ao acesso de determinado conteúdo, e também o prazo sobre esse acesso. Essa funcionalidade é chamada de leitura do contrato do VCom ou Visão sobre o VCom.

A Camada de templates é projetada a fim de fornecer à aplicação, estilos e prover uma interface ao ambiente produzido e descrito dentro do MOrFEu. Esses estilos incluem personalização de: *Layout*, *design*, navegação, e visões sobre o ambiente. Cada template criado poderá ser customizado a qualquer momento por seu dono, usando o Editor de Templates.



(A) VCom inicial, sem template **(B) VCom com template**

Figura 3.7: VCom *Blog* e uma aplicação de Templates

Ao definir VCom, o usuário usara o editor de VCom delimitando quais objetos seu espaço deve conter, quem terá acesso, por quanto tempo o acesso será dado dentre outras coisas. Na Figura 3.7.A, vemos exemplo de um *Blog* recém-criado. Finalizado o uso do Editor de VComs, o usuário terá acesso a um novo editor, o Editor de Templates, cujo produto final pode ser visto na Figura 3.7.B.

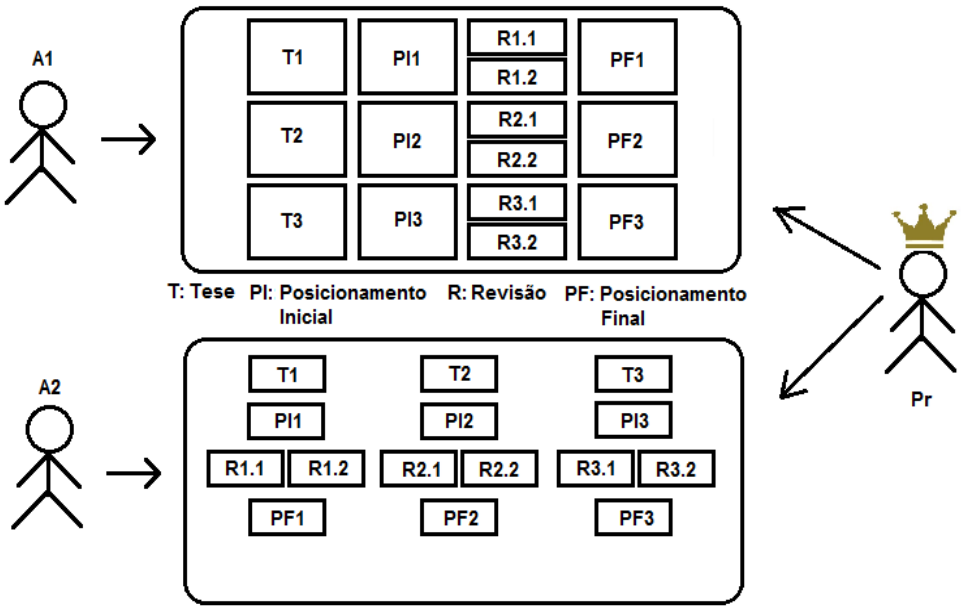


Figura 3.8: Templates com visões diferentes para usuários diferentes no debate de teses.

Além das definições gráficas de estilo de página, os templates também devem fornecer um mecanismo de interpretação das restrições de acesso impostas pelo VCom. No exemplo da Figura 3.8 usamos a arquitetura pedagógica Debate de Teses[1] [NEVADO, 2009].

Observando a Figura 3.8, no debate de teses, o personagem **Pr** insere inicialmente três teses (T1, T2 e T3) para que os atores **A1** e **A2** escrevam um posicionamento inicial (PI1, PI2 e PI3) sobre as teses. Posteriormente, outros atores, presentes ou não na imagem, podem fazer revisões (R1.2, R2.2, R2.1...) sobre esse posicionamento a fim de acrescentar ideias ou causar certo desequilíbrio nas certezas do revisado. Ao findar a etapa de revisão, A1 e A2 devem escrever seu posicionamento final sobre as teses.

Pode-se observar ainda na Figura 3.8 que A1 e A2 enxergam o espaço virtual de forma totalmente particular, e apesar desse fato, caso A1 fosse revisor de A2, A1 veria o espaço virtual como ele determinou que fosse visto, porém com as UPIs de A2. Ainda notamos que A1 e A2 apenas têm visão sobre suas teses, posicionamentos e revisões, enquanto Pr tem uma visão geral do das UPIs do ambiente.

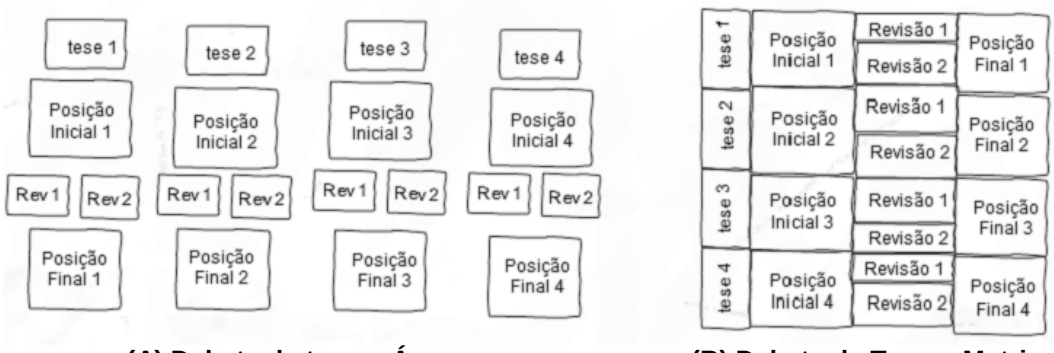
[1] Debate de testes é uma arquitetura pedagógica para discussão em grupo de variados temas. Pode ser visualizado em: http://lied.inf.ufes.br/acesso2/debateteses/dt/.

As visões, dentro do contexto do editor de templates, têm a função de dar autoridade ao dono do VCom de permitir os participantes do ambiente terem acesso às UPIs deles próprios e principalmente de outros usuários.

3.4.1 Problemas que os Templates se Propõem a Resolver

Em [GUEDES, 2009] é mencionado que a interface criativa faz surgir uma "filosofia visual" associada, inevitavelmente, à beleza. Em nossa opinião, essa nova filosofia fica mais visível, sempre que associada à eficácia e à satisfação do usuário. Entretanto, a beleza pode ser altamente subjetiva, o que faz com que o editor de templates assuma uma postura democrática.

Uma interface não atende, ao mesmo tempo, a totalidade das pessoas em potencial. Então, para que aumente a sintonia entre usuário e o ambiente, a interface precisa poder ser adaptada a ponto de não criar barreiras que venham a dificultar o uso do sistema. A diminuição dessas barreiras é o primeiro problema que o editor de templates visa transpor. Quanto mais variadas são as maneiras de efetivar uma tarefa, maiores são as chances de o usuário escolher e dominar uma delas no curso de seu aprendizado.



(A) Debate de teses: Árvore **(B) Debate de Teses: Matriz**

Figura 3.9: VCom visualizado de formas diferentes

Outro problema que pode ser resolvido com o uso de templates está descrito na Figura 3.9. Da forma como a arquitetura debate de teses foi elaborada, sua apresentação para todos os participantes restringe-se a um formato de grade, ou matriz (Figura 3.9.B) o que pode gerar desconforto cognitivo para certas pessoas. Uma melhor maneira de navegar e enxergar essa arquitetura, por

exemplo, pode ser descrita como na Figura 3.9.A, onde o debate é realizado seguindo uma estrutura arborescente.

Ainda existe o problema da acessibilidade. No contexto do espaço digital, a acessibilidade pode ser entendida como um meio de disponibilizar interfaces que atendam as preferências e necessidades especiais de cada usuário [CONFORTO & SANTAROSA, 2003]. Outra definição dada em [GUIA, 2010] diz que acessibilidade consiste em remover obstáculos indesejáveis e flexibilizar o acesso à informação e interação dos usuários que possuem alguma necessidade especial.

No MOrFEu podemos separar conjuntos de usuários por perfis. De acordo com [PIRES, 1996], a inserção de perfis para classificação de usuários de um sistema pode oferecer benefícios individuais promovendo melhores níveis de usabilidade e acessibilidade. Entretanto, esses perfis, especialmente se tratando de usuários com necessidades especiais, são afetados pela introdução de novas tecnologias, o que torna fundamental a manutenção da interação dos usuários com o sistema, além da identificação dos perfis que estão sendo afetados.

Com os templates o usuário poderá determinar a visualização do seu espaço virtual. Havendo insuficiência visual, a pessoa poderá escolher a qualquer momento o tamanho das fontes dos textos, ou então as cores do ambiente no caso de discromatopsia.

A visualização do ambiente se torna altamente flexível no que tange aos meios colocados à disposição do indivíduo, lhe permitindo personalizar a interface a fim de levar em conta as exigências da tarefa, de suas estratégias, de suas necessidades ou seus hábitos de trabalho.

3.5 CONCLUSÃO

O projeto MOrFEu busca a produção de novos espaços virtuais que sejam flexíveis desde a determinação da estrutura do espaço até a sua apresentação.

Para este trabalho, o estudo dos elementos do MOrFEu foi de suma importância uma vez que a proposta apresentada deve se encaixar no quadro geral do que se tem do meta-ambiente atualmente.

Durante a apresentação desse capítulo levantamos características estruturais que nos façam repensar sobre as propostas atuais de espaços virtuais em relação à flexibilidade que eles apresentam e na importância que isso reflete.

Discutimos sobre os componentes básicos do MOrFEu, discriminados na Tabela 3.1, enfatizando especialmente o Template e os problemas que podem ser diminuídos com sua utilização.

Tabela 3.1: Elementos centrais do MOrFEu

Elemento	Conceito
Núcleo	Responsável pela delimitação das ações sobre o MOrFEu e fornecedora de interfaces de ligação com os editores das próximos elementos dessa tabela.
UPI	Unidade de produção intelectual é o registro das composições dos participantes do ambiente virtual criado no MOrFEu. Caracteriza o conteúdo do ambiente que é completamente produzido por seus usuários.
VCom	O veículo de comunicação é a ferramenta que os participantes ou usuário do MOrFEu utilizam para determinar a estrutura do seu espaço virtual. Alguns exemplos conhecidos de VComs são: *Blog*, Fórum, *Wiki*, *Chat* etc.
Template	O template é a forma de apresentar os VComs. São responsáveis pelo estilo do espaço virtual e por identificar e pôr em prática as restrições que os VComs delimitam com relação ao acesso de seus participantes.

É importante destacar ainda nesse capítulo que o MOrFEu vem sendo projetado já há algum tempo com base em pesquisas anteriores que serviram de base para sua evolução. São elas: AmCoRA [MENEZES et al., 2000], Moonline [GAVA et al., 2001], FAmCoRA [PESSOA et al., 2002], Timoneiro [COELHO et al., 2001], AVAUFES [CAMPANA et al., 2008], o núcleo do MOrFEu [VIEIRA, 2011], a ontologia de espaços virtuais flexíveis [PERUCH et al., 2010], a modelagem e edição de VCom [RANGEL et al., 2010] e a prova do funcionamento da tecnologia XML/XSLT em ferramentas para atender as características de diferentes atividades colaborativas [SANTOS, 2010].

CAPÍTULO 4 PROPOSTA PARA O TRATAMENTO DE TEMPLATES

Construir interfaces agradáveis significa disponibilizar estímulos visuais (cores, formas, texturas e fontes) de maneira equilibrada, com o intuito de não saturar a visão e nem sobrecarregar a capacidade de assimilação das informações manipuladas pelos usuários.

O Editor de templates foi desenvolvido sob as premissas de manipulação direta [ORLANDO, 2006] e interfaces exploráveis [SHNEIDERMAN, 1983] sob a intervenção do paradigma WYSIWYG [ORLANDO, 2006], valorizando o conceito de usabilidade e proporcionando ao sujeito a opção de inventar e descobrir a melhor opção de visualização dos ambientes colaborativos criados no meta-ambiente MOrFEu.

Neste capítulo apresentaremos nossa concepção sobre o editor de templates e sua utilização na criação, edição e montagem das interfaces visuais. Dentre outras coisas, demonstraremos a arquitetura geral do sistema proposto, as etapas da customização do ambiente e os objetivos de cada uma delas.

4.1 CARACTERÍSTICAS

As características do editor foram identificadas a partir das necessidades comuns aos sistemas editores/gerenciadores de interface visual na web seguindo as indicações das abordagens de modelagem de hipermídias citadas na Seção 2.2 desse estudo.

Na Tabela 4.1 podem-se observar as principais características extraídas do estudo sobre modelagem de hipermídias orientadas a objeto. Essas características formam a base para a construção do editor de templates e são usadas na formalização dos conceitos de elaboração da solução para flexibilidade na camada de apresentação dos ambientes criados no MOrFEu.

Tabela 4.1: Características do Editor de Templates extraídas de abordagens de modelagem de hipermídias

Abordagem	Característica
OOHDM	Utiliza conceitos para descrição de navegação abstrata e composição do *layout* de apresentações a partir de elementos abstratos da interface.
WebML	Não inclui um modelo específico para expressar apresentação no nível conceitual, o fazem por meio de folhas de estilo XSLT. Utilizar XSLT para definir interface é uma boa alternativa para manter conteúdo e *layout* estritamente separados.
UWE	Uma de suas etapas inclui o processo de manutenção da hipermídia, originando um constante processo de reconstrução da interface, centrado no usuário.
OOWS	Define um conceito de visões sobre a navegação independente para cada usuário. Isso é importante para o processo de personalização da hipermídia
OOH	Valoriza o processo de autoria, o que leva a um aprimoramento da usabilidade e acessibilidade do usuário sobre o ambiente.

Duas características em especial têm foco maior sobre este trabalho. Primeiro (destacado pela abordagem WEbML), a separação da camada de apresentação em relação à estrutura e conteúdo que é o alvo principal do estudo realizado em [KERER, 2000] e, segundo (destacado em OOHDM), a percepção sobre quais elementos de interface deveriam ser afetados pelo editor. Podemos encontrar um estudo cujo foco é a detecção desses elementos em [KIRDA, 2000].

No sentido da independência entre *layout* e conteúdo, existem os padrões XML e XSL que visam proporcionar essa separação. Já a identificação das propriedades da interface visual envolve a qualidade de aspectos gráficos do texto, *layout* de página e navegação. Esses elementos estão propensos a mudar em qualquer instante para adaptação do indivíduo ao ambiente.

A partir do estudo das modelagens de hipermídias citadas e da identificação dos elementos passíveis de edição, foram destacados os requisitos para compor as principais propriedades do Editor de templates:

- Não podemos esperar que os indivíduos operantes no sistema estejam aptos a adaptar a interface às suas necessidades ou preferências no nível de codificação, mas devemos fornecer a eles meios para construção da aparência do ambiente;
- Prover total separação entre conteúdo e interface visual, o que facilita a recriação da aparência do ambiente em cada edição;
- Fornecer opções para alterações básicas sobre os aspectos gráficos das UPIs, como por exemplo: cor e tamanho de fonte, cor de plano de fundo, espaçamento do texto, margens, bordas, segregação de texto e imagem, tamanho da imagem etc.
- Permitir aos participantes do ambiente determinar a disposição dos elementos na página, modificando a visualização dos componentes da estrutura, e definir estilo de navegação sobre as páginas do espaço virtual.
- Permitir visões totalmente particulares das produções próprias e de outros usuários, dependendo do acesso concedido.

4.2 VISÃO GERAL SOBRE A ARQUITETURA

O editor foi projetado de maneira que seja de uso simples, exigindo o mínimo de esforço cognitivo, uma vez que sua interface é representada por uma linguagem gráfica de alto nível.

Com o frequente uso do editor pelos usuários, poderá ser concebida a biblioteca de templates, na qual o elemento primordial é a identificação de estratégias de organização de forma a privilegiar o reuso. Nessa biblioteca será possível a criação de templates que poderão ser especializados e instanciados com diferentes propósitos.

A criação da interface se dará sob atuação do editor nos elementos pertencentes ao VCom. Os elementos da página deverão estar dentro de containers os quais poderão ser passíveis de navegação.

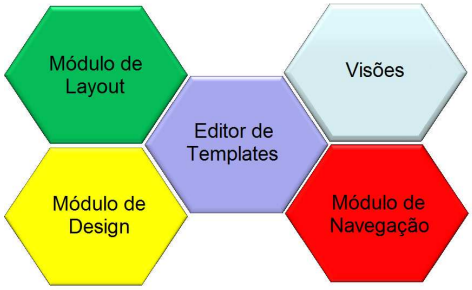


Figura 4.1: Abstração do Editor de Templates e seus módulos

Para a criação da interface, o indivíduo terá que passar por três módulos (que podem ser vistos na Figura 4.1) pertencentes à arquitetura: módulo de *layout*, módulo de *design* e módulo de navegação. Os módulos são independentes e ao final da utilização de cada um, por meio de interface própria, o template ganhará uma nova característica.

4.2.1 Módulo de *Layout*

A palavra *layout* remete a modo de distribuição e arranjo dos elementos gráficos num determinado espaço ou superfície[2]. No editor de templates chamamos então, a disposição dos elementos na tela de *layout*.

Na Figura 4.2, pode-se observar um exemplo de alteração de *layout* em um veículo que se assemelha a um *Blog*, cujos elementos que compõe o container navegável do VCom seriam título, postagem e comentários.

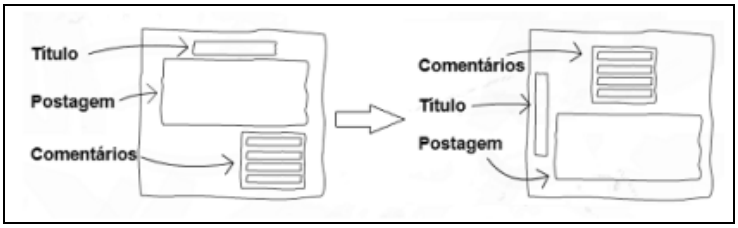


Figura 4.2: *Layout* de *Blog* que pode ser personalizado em um template no MOrFEu

A definição do *layout* é o primeiro passo da customização da aparência do VCom. O participante do ambiente terá a sua disposição uma interface com a qual deverá interagir movendo e redimensionando os objetos pertencentes ao veículo de comunicação.

---

[2] Buscamos a definição de "*layout*" (pt. Leiaute) no dicionário web: <http://www.priberam.pt>

As informações referentes à definição do *layout* serão armazenadas em folhas de estilos *eXtensible Stylesheet Language* (XSL). Posteriormente deverão ainda passar por uma transformação *eXtensible Stylesheet Language Transformations* (XSLT) para a geração da interface com usuário.

4.2.2 Módulo de *Design*

O termo *design* é usado para descrever a forma como são elaboradas obras gráficas que sejam ao mesmo tempo funcionais e estéticas[3]. No editor de templates chamamos de alterações de *design* as alterações gráficas (no sentido de desenho) sobre os objetos da interface.

A criação do *design* do editor de templates engloba as definições de: Cor de texto, cor de fundo, bordas (estilo, cor e espessura), margens, fontes (tipo, tamanho, decoração), espaçamento, identação e posicionamento de texto, imagens de fundo, dentre outras.

Uma vez definido o *design* da página, as informações gráficas serão armazenadas em folhas de estilo *Cascading Style-Sheet* (CSS) e posteriormente serão aplicadas à página pelo próprio navegador do usuário.

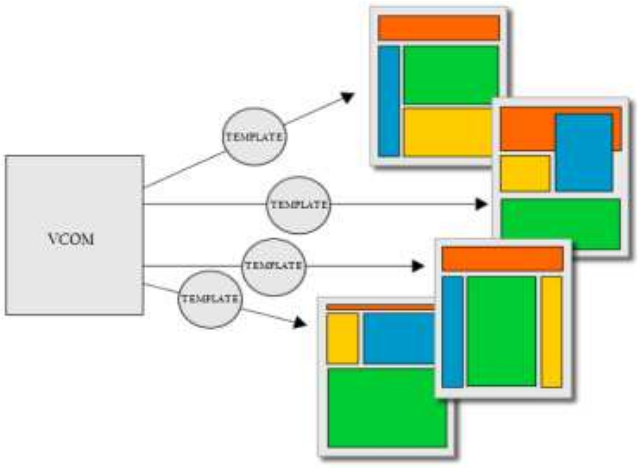


Figura 4.3: VCom com *layout* e *design*

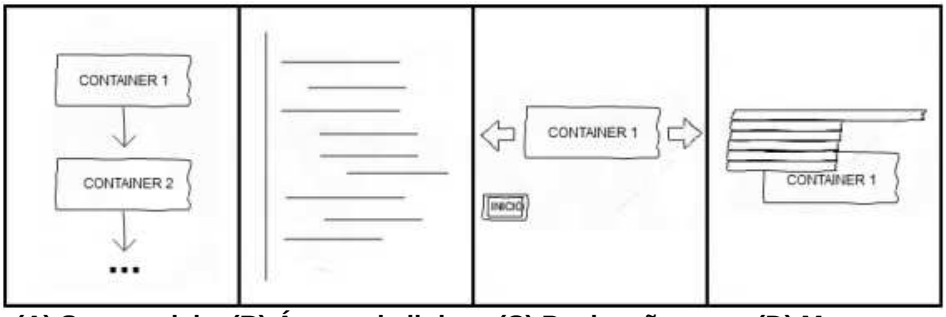
Ao finalizar o uso dos módulos de *layout* e *design*, a caracterização da interface estará quase pronta. Como se pode observar na figura 4.3, temos quatro

[3] Buscamos a definição de "*design*" (pt. Dezaine) no dicionário web: <http://www.priberam.pt>

templates diferentes elaborados para um mesmo VCom. Os containers já estão com seus objetos definidos. O próximo passo é usar o módulo de navegação para a definição da transição dos containers se estes forem passíveis de navegação.

4.2.3 Módulo de Navegação

De acordo com [WINCKLER & PIMENTA, 2000], a dificuldade mais relatada por usuários da web em encontrar informações em um site é, sem dúvida, o problema de usabilidade. Isto está relacionado à maneira como a navegação do site em questão foi projetada.



(A) Sequencial (B) Árvore de links (C) Paginação (D) Menus

Figura 4.4: Formas de navegação em VComs

Existem quatro formas principais de efetuar navegação: *i)* sequencialmente (Figura 4.4.A). *ii)* por meio de uma árvore de links (Figura 4.4.B). *iii)* transição de páginas por meio de paginação (Figura 4.4.C), ou *iv)* por meio da utilização de menus (Figura 4.4.D).

Como podemos observar na Figura 4.4, o elemento central é o container. Usando o debate de teses, na definição do VCom o usuário delimita por exemplo que os participantes do ambiente trabalharão com dez teses. Neste caso, serão dez containers que conterão uma tese, um posicionamento inicial, revisões e um posicionamento final. Cada transição de página deve carregar consigo os elementos dos containers.

A definição da estrutura de navegação será descrita em arquivos do tipo *eXtensible Markup Language* (XML). As informações armazenadas nesse documento serão necessárias para a montagem da apresentação.

4.2.4 Visões sobre o Ambiente

A arquitetura do editor de templates deverá prover mecanismos que atuem sobre a forma como os participantes enxergam o ambiente. A este passo, demos o nome de leitura de contrato do núcleo do MOrFEu.

As permissões sobre escrita e leitura de conteúdos deverão ser definidas na construção/edição do veículo de comunicação e o núcleo do MOrFEu deverá se encarregar armazenar e delimitar essas informações, gerando a figura do contrato, que determinará qual usuário poderá ler/escrever UPIs em um prazo determinado.

O papel do template envolve interpretar o contrato e tomar a decisão de mostrar ou não as produções e os botões de edição/criação de UPIs, dependendo do vencimento do prazo e do participante em questão. Mais além, o template tem a função de apresentar o conteúdo descrito por outros participantes sob a visão particular determinada por seu usuário.

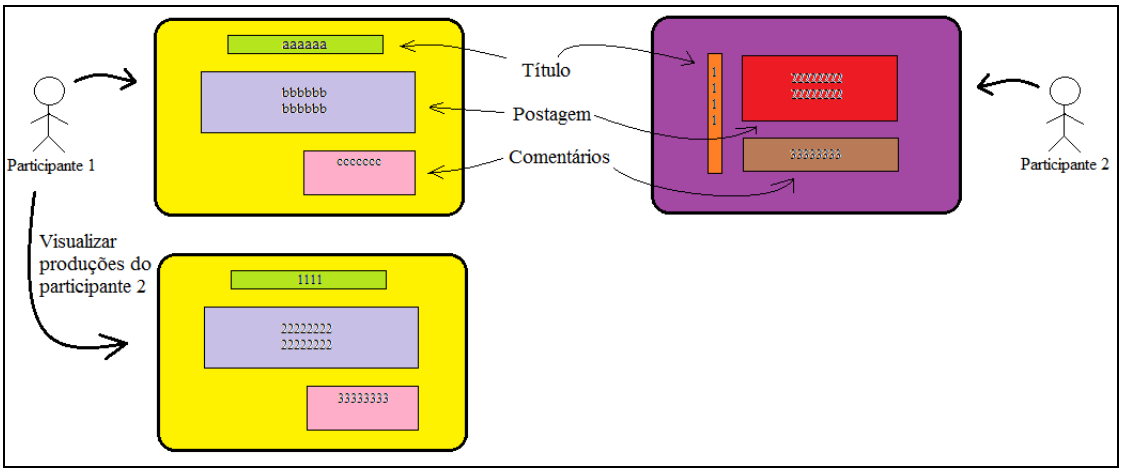


Figura 4.5: Exemplo do conceito de visões

Um exemplo sobre o uso das visões, podemos encontrar na Figura 4.5, onde o participante 1 determinou o seu template completamente diferente do template determinado pelo participante 2 e mesmo assim ao visualizar as produções do participante 2, tem uma visão particular (como ele mesmo definiu) do ambiente. Podemos dizer que o participante 1 poderá acessar as produções de outros usuários com a visão dele.

O usuário ainda tem a opção de ver o ambiente de outro participante com o template original, isto é, o template do participante acessado. Essa situação ocorre principalmente se o usuário que está acessando for anônimo (visitante).

Em um segundo exemplo, imaginemos a situação do professor em relação ao aluno no VCom Debate de Teses. Nesse caso, o usuário professor pode criar teses e pode ver as produções de todos os alunos (posicionamento inicial sobre a tese, por exemplo), mas não pode editá-las. Já os alunos estão restritos a ver apenas suas próprias produções e as revisões feitas sobre elas (que seriam produções de outros alunos) e só podem editá-las em um prazo determinado pelo professor. O professor então tem uma visão muito mais abrangente do espaço virtual do que seus alunos, e os alunos terão sua visão sobre o ambiente, modificada com o tempo, uma vez que a opção de editar as produções em certo momento, desaparecerá.

4.3 FUNCIONAMENTO DO EDITOR DE TEMPLATES

O editor de templates funciona em dois momentos distintos, durante o desenvolvimento da interface gráfica (quando o usuário determinar *layout*, *design* e navegação), e durante a execução das tarefas no espaço virtual (enquanto a arquitetura lê o contrato das visões).

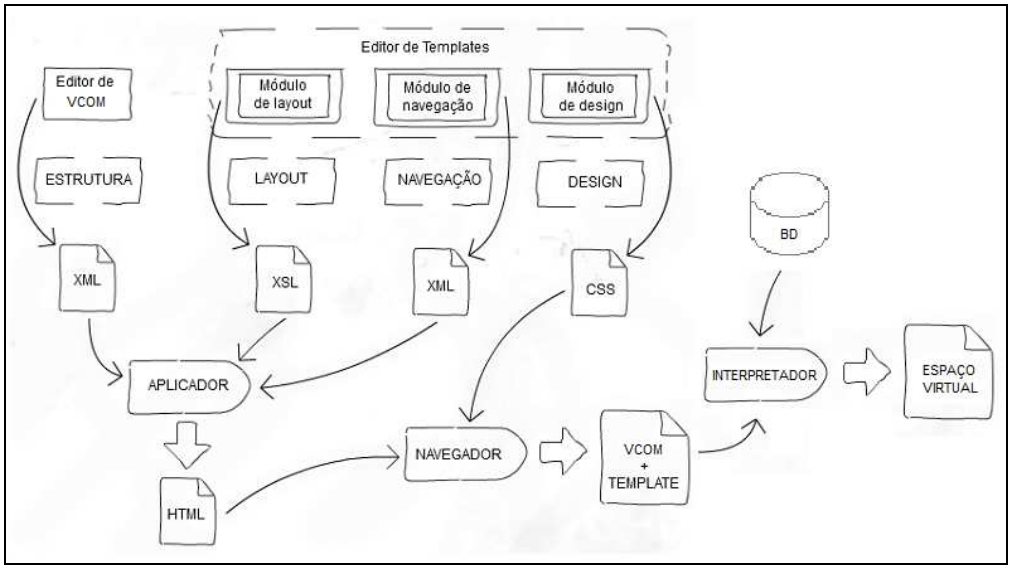


Figura 4.6: Visão Geral do Editor de Templates

Na Figura 4.6 podemos observar a disposição dos módulos, os documentos gerados em cada fase, e as estruturas programáticas que executam o processo de transformação dos documentos gerados. Baseando-se nela, explicaremos como se dá o funcionamento do editor. Informações sobre as tecnologias utilizadas no processo, como XML, XSL e CSS são mais bem descritas no Apêndice I.

Inicialmente, o ambiente estará descrito em um documento XML bem estruturado que contém todas as características do VCom. Algumas informações importantes que deverão ser carregadas nesse documento são, por exemplo: número total de containers que poderão existir no espaço virtual, número de respostas que poderão ser efetuadas para uma postagem, o título do espaço virtual, dentre outras coisas.

Em todas as etapas do processo da construção da interface visual do ambiente, os usuários poderão contemplar uma pré-visualização da interface desejada do ambiente, obedecendo ao paradigma WYSIWYG.

O primeiro passo na criação do template é a definição do seu *layout*. O indivíduo terá acesso a uma interface do tipo *drag'n drop/resize* que o possibilitará "desenhar" sua página, isto é, escolher a posição dos elementos do espaço virtual na tela. Definido o *layout*, um documento XSL será gerado, documentando essa estrutura.

O próximo passo é a definição do *design*. Ao salvar as configurações do *layout*, o usuário será direcionado a uma segunda interface que fornecerá artefatos para a definição gráfica dos elementos. Essas definições são baseadas nas propriedades de classes em CSS. Neste caso, as alterações de *design* suportadas pelos templates e oferecidas pelo editor, são as mesmas fornecidas pelas definições CSS.

No terceiro e último passo o indivíduo será encaminhado a uma nova interface que o permitirá escolher o estilo de navegação do ambiente, e a opção será

salva na base de dados. Ainda nesse módulo, o usuário poderá publicar seu template em um próximo passo.

O mecanismo de publicação funciona como um ajuntador de informações. Na Figura 4.6 ele é denotado como o **aplicador**. Os documentos gerados nas fases de descrição de *layout* e navegação, são aplicadas ao VCom gerando um documento HTML que é o padrão dos navegadores da web. Posteriormente o documento do *design* será aplicado ao HTML pelo próprio navegador do indivíduo. No final desse processo, o veículo de comunicação já estará publicado com a interface visual determinada pelo seu participante.

O template poderá ser reutilizado por outro participante do ambiente no qual ele foi aplicado. Havendo a necessidade ou a vontade de alguma alteração, os arquivos gerados serão alterados conforme as mudanças realizadas. Se a alteração em questão for uma alteração de *design*, um novo documento CSS será gerado conforme as mudanças. Caso a alteração seja no *layout*, o documento recriado será o XSL e o mesmo vale para a navegação. Então ao conjunto dos arquivos XSL, CSS e XML, damos o nome de template gráfico.

A partir do momento em que o template gráfico entra em ação, é necessário um novo mecanismo que atue em tempo de execução para ler a determinação do núcleo do MOrFEu com relação ao acesso das UPIs dentro do veículo de comunicação. Na Figura 4.6, esse mecanismo é representado pelo **interpretador**, que se encarregará de montar as visões sobre o ambiente a partir da leitura do contrato do núcleo. O contrato é criado sobre as regras delimitadas no VCom. Nesse ponto, o interpretador mostrará ou não as UPIs e permitirá ou não edição sobre elas.

Portanto, o template é formado pela interface gráfica em conjunto com as visões sobre o ambiente, sendo que o primeiro é determinado em tempo de desenvolvimento da interface, enquanto o segundo é determinado em tempo de execução.

4.3.1 Estrutura da linguagem de descrição de VCom e a relação dele com o Template

Uma proposta de definição de VCom foi elaborada em [RANGEL, 2011] e está sendo usada para definir o veículo de comunicação neste trabalho. Mas ainda precisávamos formalizar a estrutura de descrição dos veículos para aplicação dos templates, então usamos a linguagem XML para criar um vocabulário cujas principais marcações estão descritas na Tabela 4.2.

Tabela 4.2: Enumeração das principais marcações da linguagem de descrição de VComs

Marcação	Descrição
<CONTAINER/>	É um agrupador de elementos, e é a marcação que determinará se os elementos em seu interior passarão por alguma transição
<AMOUNT/>	É a marcação que define a quantidade de containers navegáveis que um ambiente poderá possuir.
<TERM/>	Define o prazo de leitura/escrita sobre um determinado elemento do espaço.
<AWNSERS/>	Determina a quantidade de postagens que um determinado elemento deverá ter.

Todas as informações referentes ao ambiente estarão descritas e definidas no documento XML do VCom, incluindo as definições de leitura e escrita (que serão usadas em larga escala no processo de execução das visões) dos usuários que participarão do ambiente.

Na Figura 4.7 temos um exemplo de descrição de um *Blog* bem simples utilizando a linguagem que foi elaborada. Este espaço virtual, a partir da definição de seu template, terá um título fixo, e dez (<AMOUNT>) containers com elementos que vão passar por transição de páginas.

Pode-se observar que os elementos títulos e postagem terão prazo (<TERM>) de dois dias a partir da data de criação do VCom para serem lidos e/ou escritos. Já para os comentários o prazo será de apenas um dia. Repare ainda que para cada título e postagem, poderá haver até três comentários (<AWNSERS/>).

```
<VCOM>
    <HEADER>
        ...
    <HEADER/>
    <VCOMSTRUCT>
        <CONTAINER NAVIGATION=FALSE>
                <TITLE>
                    <TEXT>BLOG</TEXT>
                </TITLE>
        </CONTAINER>
        <AMOUNT>10<AMOUNT>
        <CONTAINER NAVIGATION=TRUE>
                <t>
                    <TERM>2</TERM>
                    <AWNSERS>1</AWNSERS>
                    <NAME>TITULO</NAME>
                </t>
                <P>
                    <TERM>2</TERM>
                    <AWNSERS>1</AWNSERS>
                    <NAME>POSTAGEM</NAME>
                </P>
                <C>
                    <TERM>1</TERM>
                    <AWNSERS>3</AWNSERS>
                    <NAME>COMENTÁRIOS</NAME>
                </C>
        </CONTAINER>
    </VCOMSTRUCT>
<VCOM/>
```

Figura 4.7: Descrição simples de um *Blog*

É importante destacarmos a forte dependência entre Templates e VComs. Um veículo de comunicação sem um template é apenas um documento sem utilidade para os usuários comuns. É um espaço virtual formalmente descrito, que nunca poderá ser utilizado.

Um VCom pode existir sem um template mas, necessariamente, para ser apresentado, precisa de um. O contrário não se aplica, isto é, um template, para existir, necessariamente precisa estar atrelado a um VCom, ou mais de um. Isto significa que um mesmo template pode ser usado para diferentes instâncias de um mesmo veículo de comunicação.

4.4 MODELAGEM DA PROPOSTA

Nesta seção apresentaremos a modelagem da proposta para o editor de templates. Inicialmente discutiremos alguns casos de uso, posteriormente

relataremos o diagrama de classes proposto e, em seguida, apresentaremos o fluxo de atividade que esperamos obter.

A Figura 4.8 representa o esquema de diagrama de casos de uso, com maior ênfase nas ações sobre os templates no MOrFEu. Temos três atores principais, o dono do VCom, ou administrador, o participante e o visitante ou anônimo. Todos os atores necessariamente executam o caso de uso **acessa espaço virtual**, porém nem todos precisam passar pela análise do contrato. Este caso de uso deverá ser executado pelos atores visitante e participante.

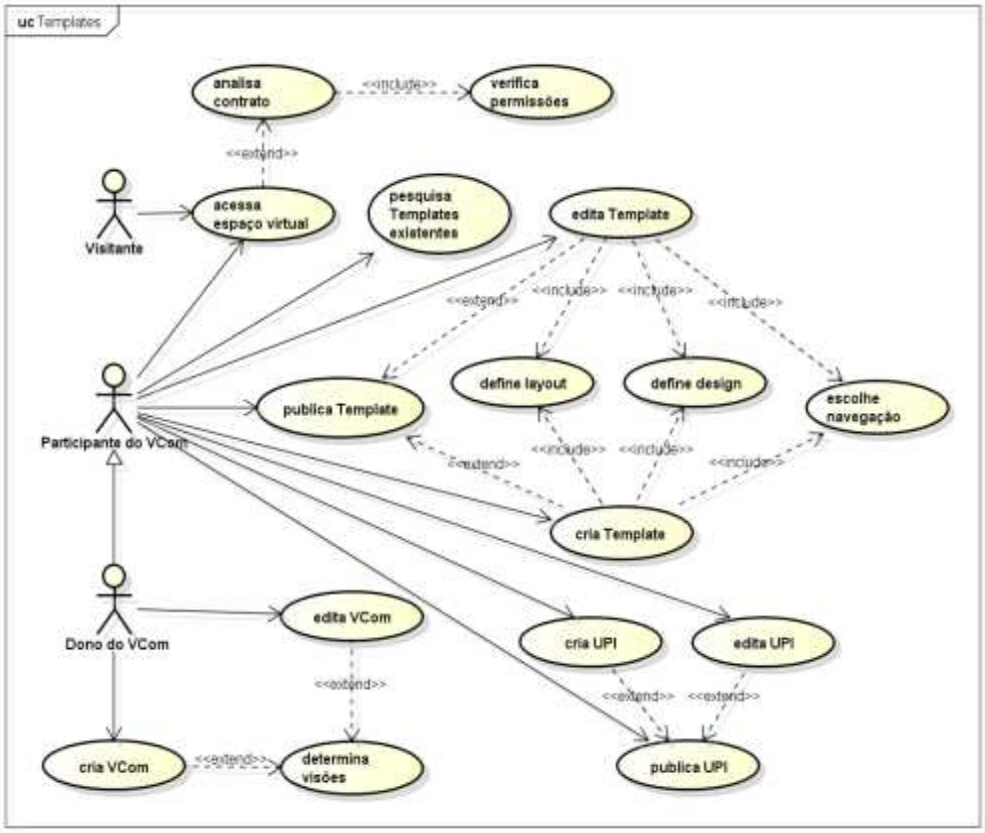


Figura 4.8: Diagrama de casos de uso com ênfase nos templates

O participante e o dono do VCom, poderão pesquisar templates previamente desenvolvidos a fim de terem uma base para seus próprios templates. Essa funcionalidade é descrita pelo caso de uso **pesquisa Templates existentes**. Esse caso de uso essencialmente deve ser realizado se o caso de uso **edita Template** for executado.

Com base na Figura 4.8, identificamos que a realização dos casos de uso **cria Template** e **edita Template** necessariamente faz com que ocorra a execução

dos casos de uso **escolhe navegação**, **define *layout*** e **define *design***, ou seja, ao cria/editar template necessariamente é necessário executar os três passos. Contudo, um template criado/editado, não necessariamente precisa ser publicado. Essa etapa pode realizada a qualquer momento.

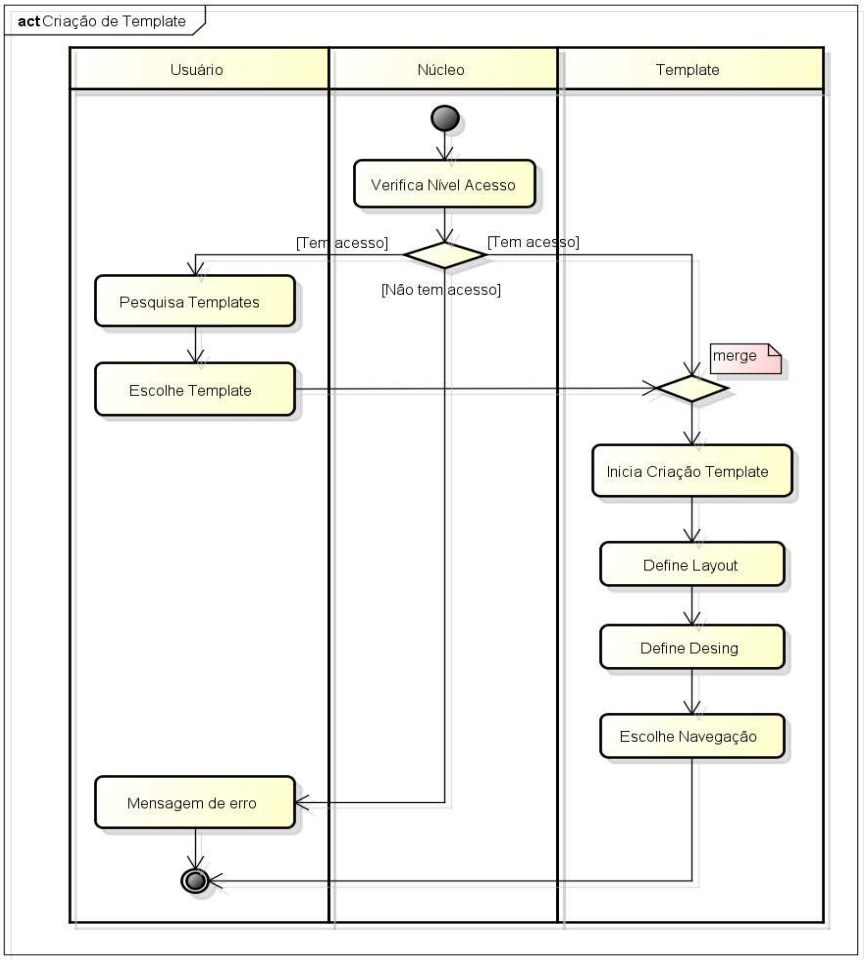


Figura 4.9: Diagrama de Atividade - Criar Template

A fim de explicar melhor a realização do caso de uso **cria Template**, desenvolvemos um diagrama de atividade que está descrito na Figura 4.9. Inicialmente precisa-se identificar o tipo de usuário que acessa o editor no momento. Essa informação o núcleo já tem armazenada. Caso o usuário seja um visitante, ele não pode criar templates. Então é mostrada uma mensagem de erro e é encerrada a atividade. Caso o usuário seja um participante ou administrador, ele escolhe criar um novo template ou pesquisar templates existentes e usá-los. Ainda existe uma terceira opção que não está descrita no diagrama, que seria modificar um template existente.

A partir do início da criação do template, o usuário necessariamente precisa definir o *layout* e *design*, e escolher como navegar entre os containers do template. O diagrama também vale para o caso de uso **edita Template**.

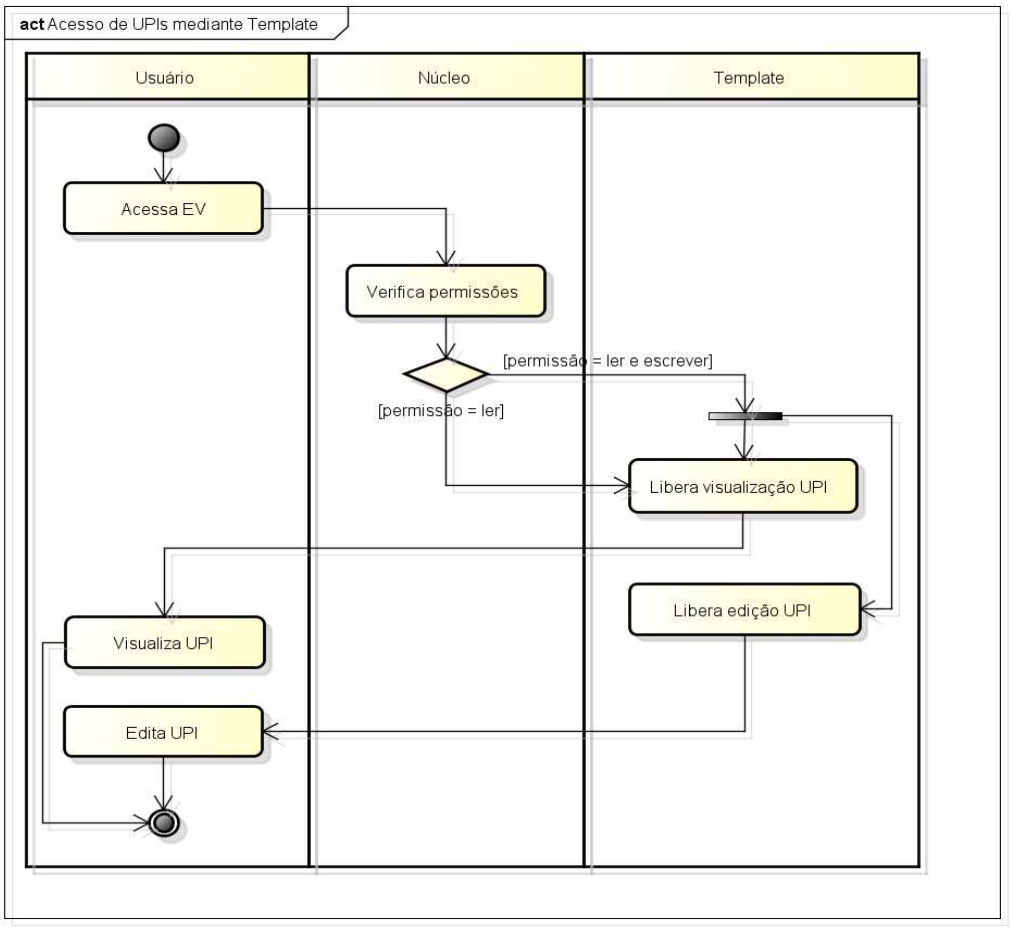


Figura 4.10: Diagrama de Atividade - Templates e UPIs

O caso de uso **Acessa Espaço Virtual** depende estritamente que as permissões concedidas sobre as UPIs estejam muito bem definidas na elaboração do VCom. O núcleo se encarregará de armazenar as informações sobre as permissões, montando assim a imagem do contrato.

Como pode ser visto na Figura 4.10, o template receberá as informações do contrato (se o usuário em questão tem permissão de leitura e escrita de somente leitura ou nenhuma) e se encarregará de mostrar as UPIs para visualização ou de mostrar o botão para edição das produções.

4.4.1 Diagrama de Classes

O diagrama de classes desenhado na Figura 4.11 é um fragmento do modelo que descreve o MOrFEu. Nele podemos observar alguns detalhes essenciais para o entendimento das operações no MOrFEu.

No modelo elaborado, existem três tipos de usuário: Participante, Administrador e Visitante, sendo que todos podem acessar um VCom especializado de acordo com seu nível de acesso. Contudo, somente poderão criar templates os usuários administrador e participante, e apenas o administrador pode especializar um VCom.

Os templates deverão ser parte constituinte do veículo de comunicação especializado, porém eles somente poderão existir se um VCom já tiver sido especializado. No diagrama, observa-se ainda que toda produção (UPI) deverá ser mostrada seguindo as restrições (permissões) de um contrato.

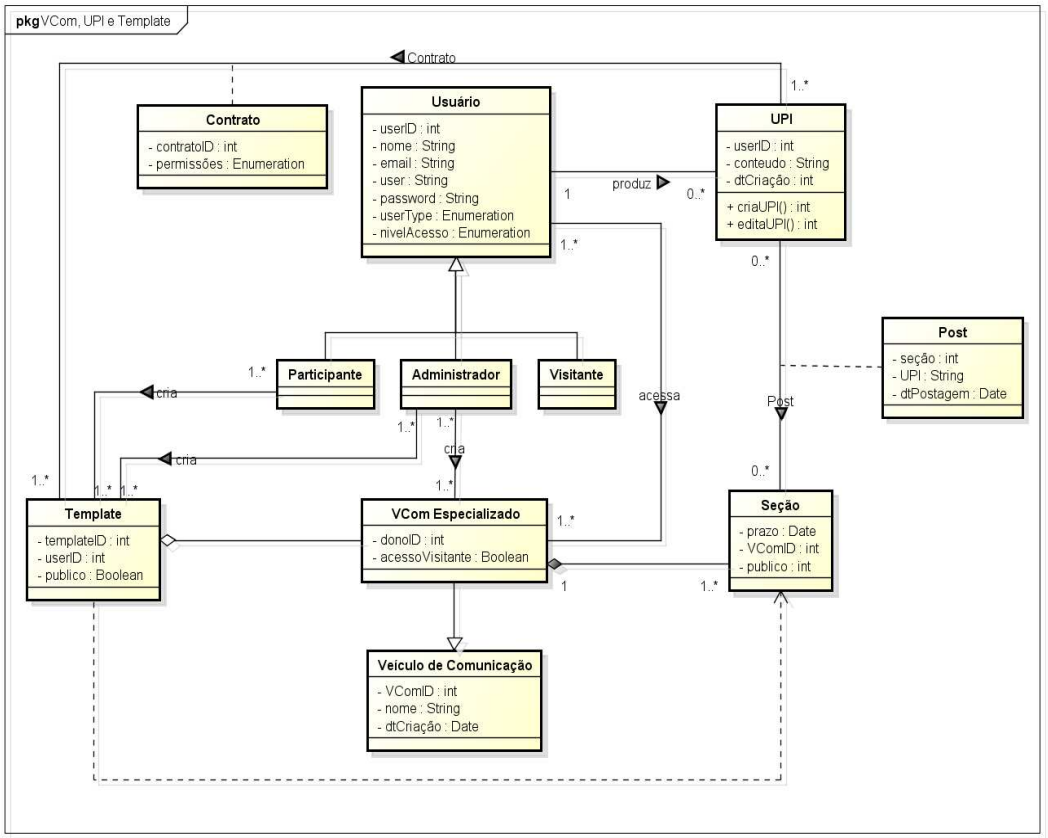


Figura 4.11: Diagrama de classes envolvendo Templates, UPI e VCom

O VCom especializado compõe a seção, e para um template existir, é necessário a existência do VCom especializado. Isto faz com que exista uma relação de dependência entre o template e a seção. No diagrama ainda nota-se que toda UPI é adicionada à seção por meio de um post, ou postagem, contudo, o post não necessariamente precisa existir uma vez que a cardinalidade de ambas as classes da relação podem ser zero.

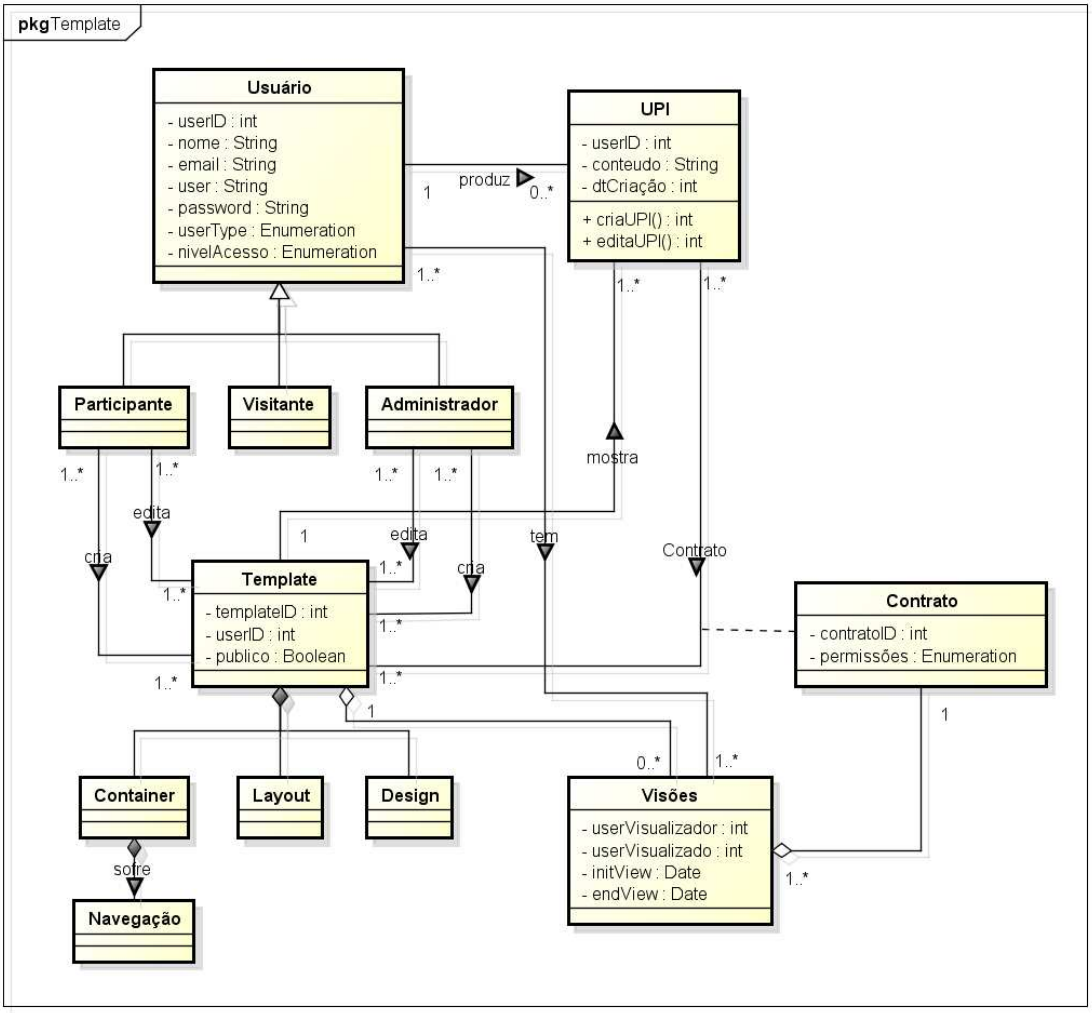


Figura 4.12: Diagrama de classes com ênfase nos templates

Os templates serão sempre compostos de Navegação, *Layout* e *Design*, podendo ter ou não Visões definidas para ele. No modelo da Figura 4.12, podemos ver que as visões somente poderão existir caso um contrato tenha sido estabelecido e devem determinar os participantes que visualizarão as UPIs de outros participantes e o prazo para que isso ocorra.

A classe Contrato está em um nível acima da classe Visões. As visões são estabelecidas com base nas permissões que o contrato oferece e afetam diretamente os usuários. Já o contrato não se envolve com o usuário, mas estipula que tipo de permissão poderá ser fornecido quando as visões forem concedidas, estabelecendo a relação entre o template e a visualização da UPI.

4.5 CONCLUSÃO

A proposta do Editor de Templates, apresentada neste capítulo, foi elaborada sob a premissa de que interfaces flexíveis são úteis e importantes para facilitar as interações dos usuários com o ambiente virtual, seja para aprendizagem, colaboração ou simplesmente para utilização pessoal [GUEDES, 2009].

O MOrFEu em sua essência busca inserir flexibilidade nos ambientes virtuais a fim de permitir a reprodução de espaços virtuais que se moldem às necessidades de seus participantes. A flexibilidade, nesse caso, necessariamente precisa englobar todas as camadas do MOrFEu, desde produção de conteúdo à descrição da interface.

O Editor de Templates é a solução para a questão da flexibilidade na camada de apresentação dos espaços virtuais criados no MOrFEu. A customização das interfaces dos ambientes deve envolver todos os elementos de interface, incluindo até mesmo a amostragem de conteúdo para certos participantes.

Nesse capítulo descrevemos todos os elementos passíveis de customização (*layout*, *design* e navegação) e como o módulo de cada um desses elementos deve se comportar no editor mediante sua alteração. Além disso, especificamos as características do editor, que envolvem funcionalidades e comportamento.

Explicamos como se dá o funcionamento do editor e como são realizadas as interações entre os módulos do projeto, e formalizamos um resultado esperado pela a execução de cada módulo. Os resultados são representados por documentos de estilo: para o *layout*, um arquivo XSL, *Design*, um documento

CSS e para a navegação um documento XML. Esses arquivos documentam as definições de customização que usuários escolheram durante o uso do editor.

Finalmente, descrevemos uma modelagem sobre a proposta, exemplificando os conceitos, relações e ações usando definições de engenharia de software como diagramas de caso de uso, de atividade e de classes.

A formalização da proposta para o tratamento de templates no MOrFEu foi extremamente útil para elaboração do protótipo que será descrito no próximo capítulo. Os diagramas construídos nos permitiram ter uma visão muito mais ampla do conceito de templates e de como se daria seu tratamento. Além disso, serviram de base nos momentos de definição de estratégia para programação computacional.

CAPÍTULO 5 UMA IMPLEMENTAÇÃO DO EDITOR DE TEMPLATES

Este capítulo apresenta uma implementação para a proposta do Editor de Templates descrita anteriormente. O capítulo é organizado da seguinte forma: A Seção 5.1 fala brevemente sobre as tecnologias usadas na codificação do projeto. A Seção 5.2 mostra a primeira versão de um protótipo e como se dá sua utilização. Em 5.3 apresentamos as limitações da implementação e, finalmente, na Seção 5.4 escrevemos as conclusões do capítulo.

5.1 TECNOLOGIAS

Para o desenvolvimento do Editor de Templates buscamos a utilização de ferramentas e linguagens com software livre. Como o editor foi desenvolvido para funcionar totalmente na plataforma web, a programação foi dividida em cliente e servidor.

A programação da parte do servidor foi toda realizada usando a linguagem PHP, enquanto na parte do cliente usamos HTML, Javascript e CSS. Para o armazenamento em banco de dados foi utilizado o SGBD MYSQL, e como servidor web, usamos o Apache.

Além das linguagens já mencionadas, usamos XML para descrever informações de transição de página como forma de navegação e transformações XSLT para transformação de XML em HTML usando um documento XSL como base do *layout* da página. As especificações dessas tecnologias podem ser obtidas no site da W3C "*World Wide Web Consortium*", um consórcio de diversas empresas que buscam estabelecer padrões para a Internet. No Apêndice I encontramos breves descrições das tecnologias utilizadas na implementação do editor de templates.

Uma vez definidas as ferramentas e tecnologias usadas para o desenvolvimento do editor de templates, demostraremos a seguir os detalhes de implementação do editor de templates.

5.2 PRIMEIRA VERSÃO DO EDITOR DE TEMPLATES

A última parte desse estudo constituiu-se no desenvolvimento da primeira versão do Editor de Templates. A Figura 5.1 demonstra a página inicial do editor, na qual o usuário poderá acessar um espaço virtual (VCom + Template) previamente definido, criar um novo template para um VCom específico ou editar um template existente.

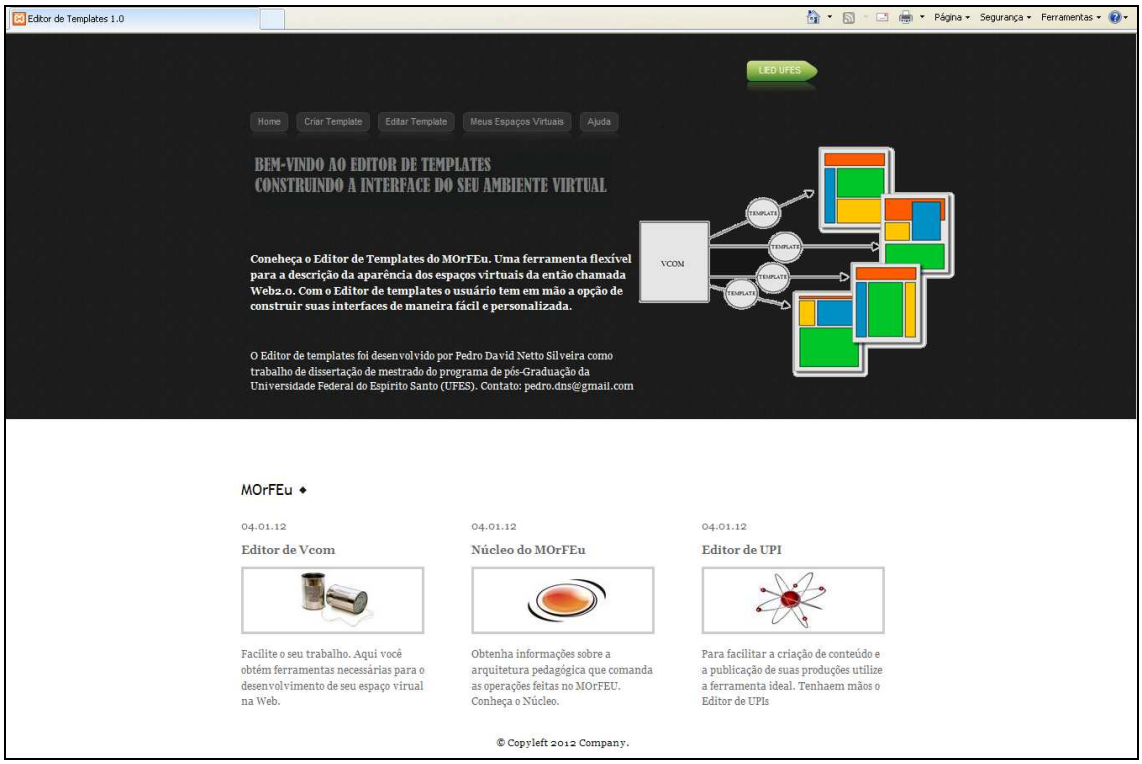


Figura 5.1: Página Principal do Editor de Templates

A seguir dividiremos a apresentação do protótipo em três partes. Primeiro, como criar um Template para um VCom específico a partir do início; segundo, como editar o template criado e, por último como acessar o espaço virtual com visões previamente definidas. Posteriormente relataremos como foi definida a estrutura de arquivos do protótipo e como funciona o mecanismo das visões sobre o ambiente.

5.2.1 Criar Template

Inicialmente, o usuário deve clicar no botão “Criar Template” da Figura 5.1. Esse botão dispara um evento que mostra a janela em destaque na Figura 5.2. Um VCom deverá ser escolhido e um nome novo deve ser dado ao template.

Figura 5.2: Escolha do VCom que receberá o template

Para este exemplo, escolhemos o VCom “*Blog*1” da Figura 5.2 e demos o nome “Template_*Blog*” ao template criado. Inicialmente, um primeiro interpretador do Editor de Templates se encarregará de ler o documento do VCom e disponibilizar os seus elementos na tela para sua edição. A Figura 5.3 mostra o exemplo de um VCom ainda sem template que assume o papel de um *Blog*.

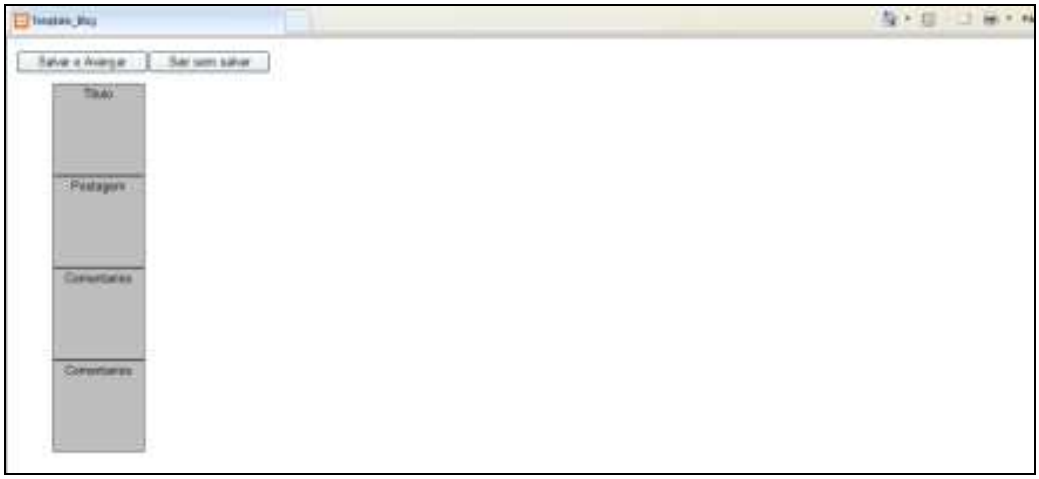


Figura 5.3: Descrição gráfica do VCom sem template

Uma vez que esses elementos estejam disponibilizados na tela do participante do ambiente ele se fartará do uso do módulo de *layout* do editor para redimensionar e reposicionar cada elemento dentro do container usando apenas o mouse. Na Figura 5.4 podemos ver um exemplo do trabalho com o primeiro módulo do Editor de Templates.

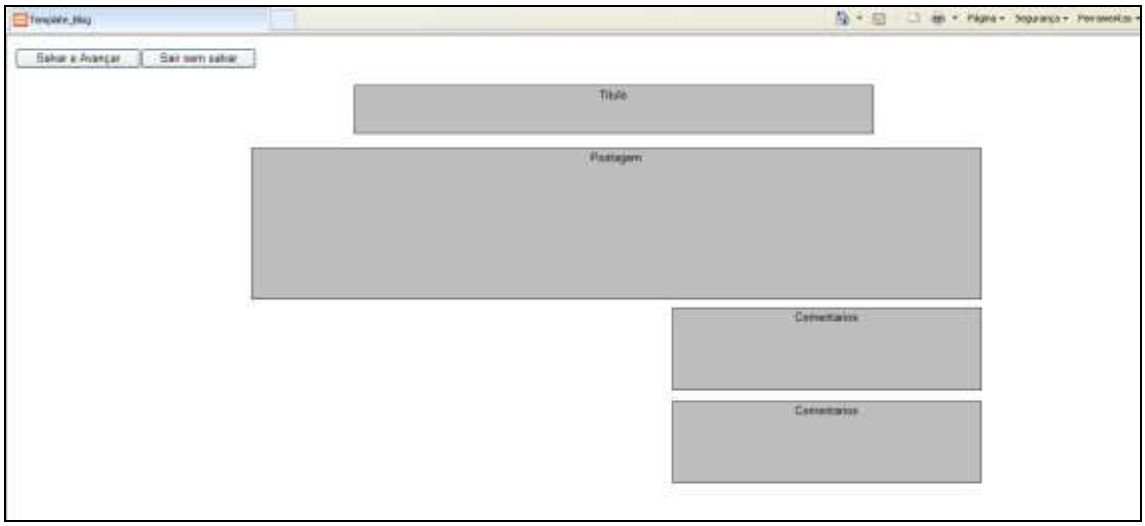


Figura 5.4: Utilização do módulo de *Layout*

Ao clicar em “Salvar e Avançar”, o usuário automaticamente será redirecionado para o próximo módulo (*Design*) e o servidor se encarregará de guardar as definições do usuário gerando um documento XSL que ficará armazenado na pasta de *layout*s salvos.

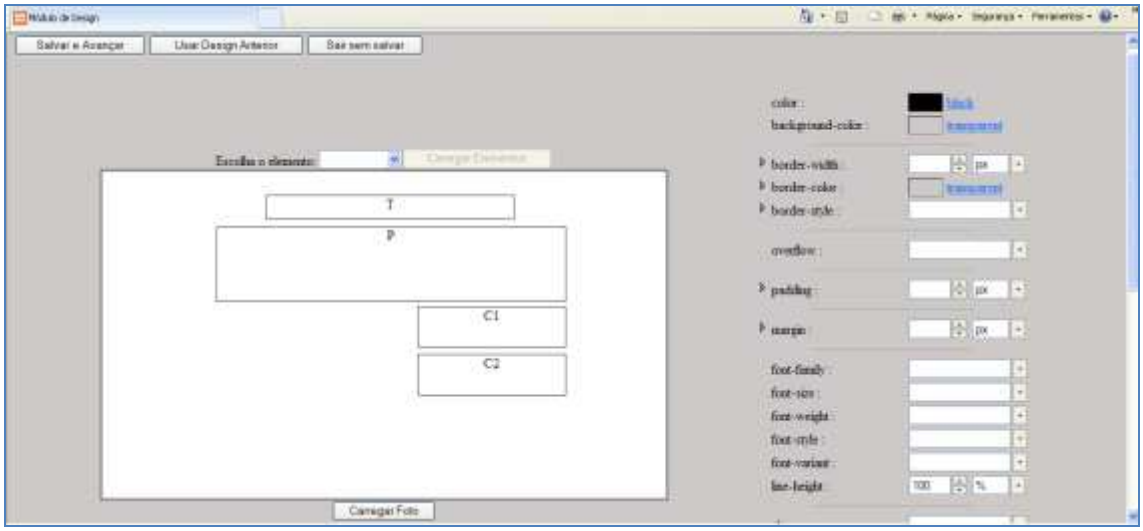


Figura 5.5: Módulo de *Design*

Na Figura 5.5 temos o módulo de *design* após o clique no botão “carregar elementos”. Será então mostrada ao usuário uma miniatura do *layout* definido

no módulo anterior para que, ao escolher as propriedades de *design*, as alterações possam ser vistas pelo usuário em tempo real.

Na Tela da Figura 5.5 ainda existe um botão chamando “Carregar Foto”, que quando acionado exibe uma tela popup para efetuação de upload de arquivos de imagens. Essa funcionalidade é útil para o usuário que queira definir uma imagem qualquer como plano de fundo, ou até mesmo como fundo de um elemento do container.

A necessidade do usuário ver exatamente como sua interface está sendo definida enquanto a constrói é essencial para a satisfação do indivíduo. Esse conceito que envolve o paradigma WYSIWYG é explorado por [ORLANDO, 2009] e é empiricamente funcional.

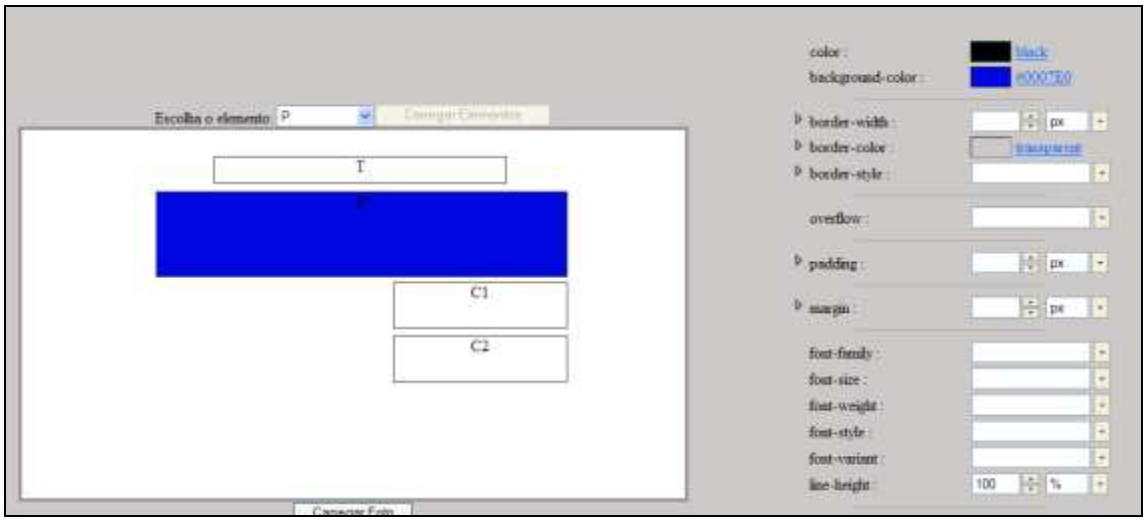


Figura 5.6: Utilização do Módulo de *Layout*

No exemplo da Figura 5.6, o participante escolheu o elemento “P” (Postagem) e definiu a propriedade “background-color” como azul, e o elemento sofreu a alteração na tela imediatamente após a escolha do usuário. Essa funcionalidade vale para qualquer propriedade do módulo de *Design*.

Para esse exemplo, ao final da utilização do módulo de *design* definimos as propriedades dos elementos do VCom, como mostrado na Figura 5.7. No próximo módulo, o usuário continuará tendo uma prévia do modelo do espaço virtual que ele espera obter.

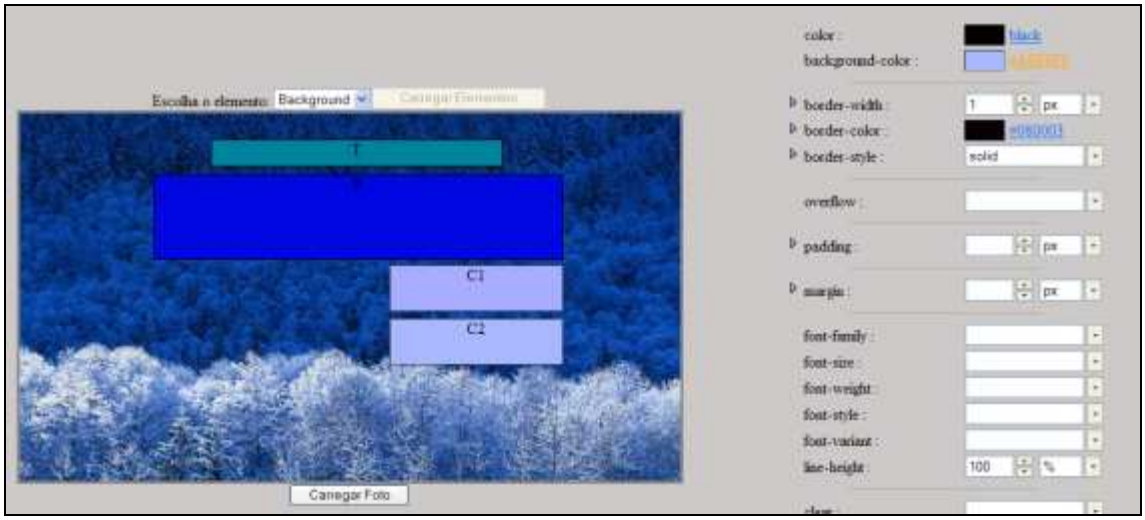


Figura 5.7: Final da utilização do Módulo de *Design*

Ao clicar em “Salvar em Avançar” (botão pode ser visto na Figura 5.5) um documento CSS será gerado documentando o *design* do ambiente e o participante será redirecionado ao próximo módulo a fim de definir como se dará a navegação entre os containers do espaço virtual. Na Figura 5.8 está representada a tela de escolha do tipo de navegação: Estrutura Sequencial, Árvore de Links, Paginação ou Estrutura de Menus.

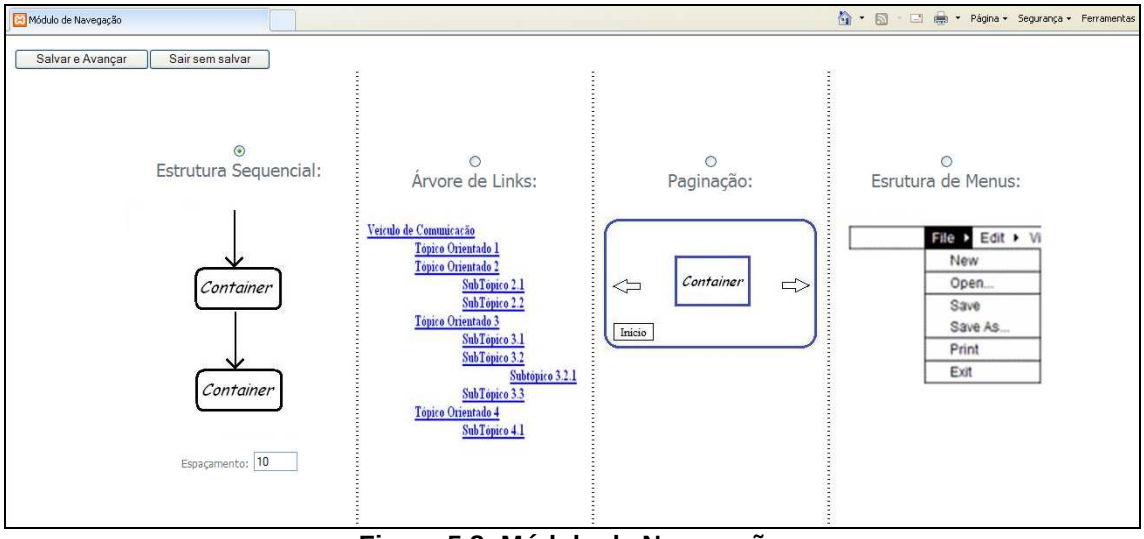


Figura 5.8: Módulo de Navegação

Estrutura Sequencial: Caso seja escolhida essa opção, não haverá transição de páginas no ambiente do participante, isto é, todos os containers principais estarão dispostos em uma mesma página (como pode ser visto na Figura 5.9) e espaçados de acordo com o valor que for mensurado no campo “Espaçamento”. Este valor é dado em quantidade de pixels (px).

Repare que na parte direita da imagem, a barra de rolagem contém três passadores, o que significa que a imagem foi dividida em três partes e remontada, a fim de mostrar que os containers principais estão em uma mesma página, e para serem vistos, é necessário que se role a página para baixo.

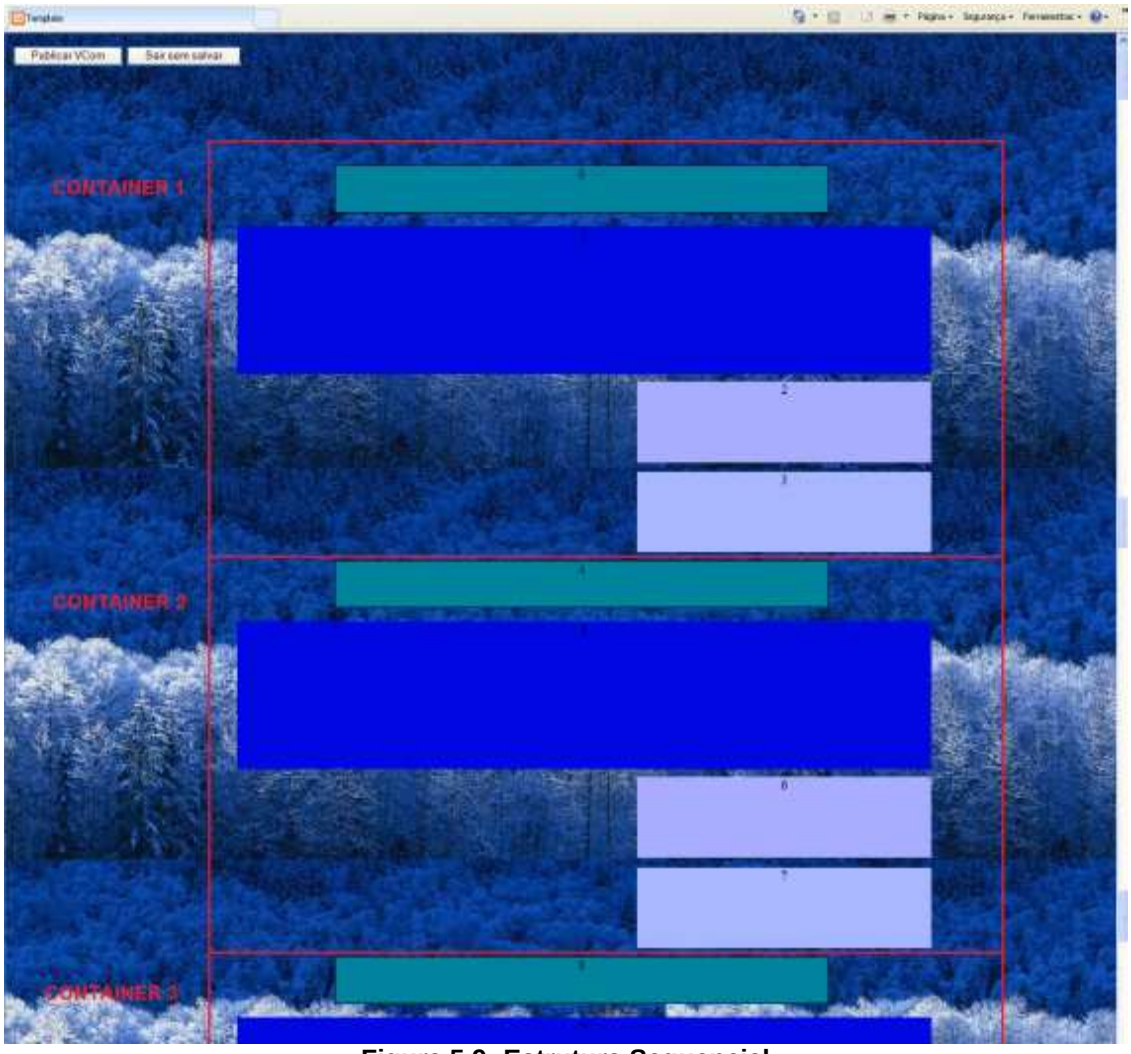


Figura 5.9: Estrutura Sequencial

A Figura 5.9 mostra uma pré-visualização para o usuário sobre como ficará o template final de seu espaço virtual. Contudo, dentro dos elementos ainda não existe nenhum conteúdo (UPI), eles contêm apenas alguns números sequenciais para diferenciar os elementos caso seja escolhida um tipo de navegação que trabalhe com transição de páginas como os que vêm a seguir.

Árvore de Links: Essa é a segunda opção de navegação que o usuário pode escolher. Dependendo do número de containers (e seus nós filhos)

especificado no documento do VCom, será montado um quadro com os links para cada container principal do template, como destacado na Figura 5.10.

Figura 5.10: Navegação - Árvore de Links

Essa opção de navegação permite acesso direto a containers específicos. A Figura 5.11 mostra como ficaria o template caso o participante acessasse o segundo link (Tópico 1) do quadro. Como podemos ver, os números dentro dos elementos seguem a sequência 4~7. Dessa forma o participante consegue identificar nessa pré-visualização da navegação que se trata do segundo container do template (uma vez que a sequência do primeiro seria 0~3).

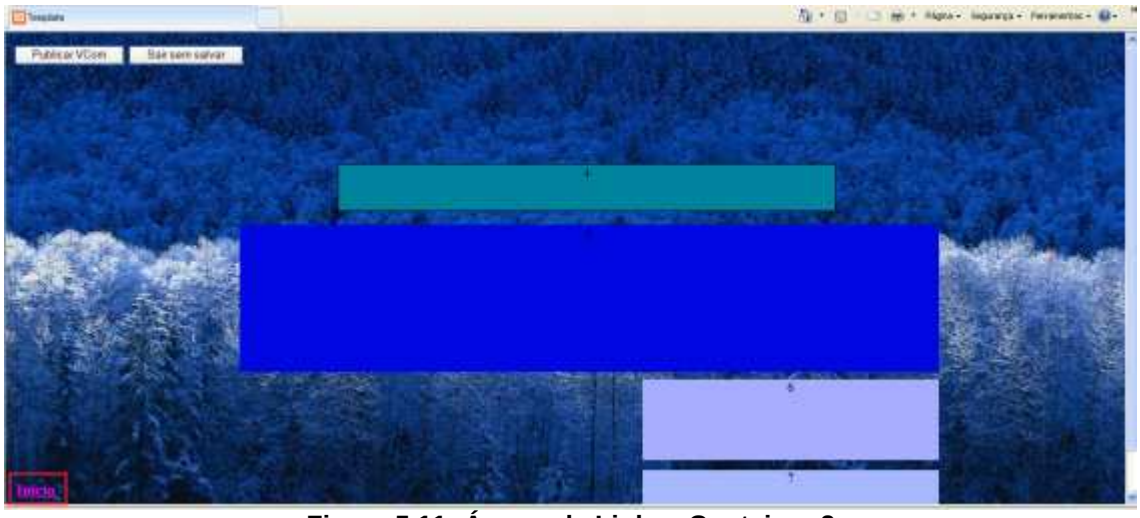


Figura 5.11: Árvore de Links - Container 2

Uma vez escolhido um link, para ter acesso ao quadro novamente, o participante interage com o template clicando no link “Inicio” como destacado na Figura 5.11. Fazendo isso, o template volta a ficar como na Figura 5.10.

Paginação: Com essa opção o usuário deverá navegar sequencialmente pelo template, interagindo com os botões “avançar”, “voltar” e “inicio”, como destacados na Figura 5.12

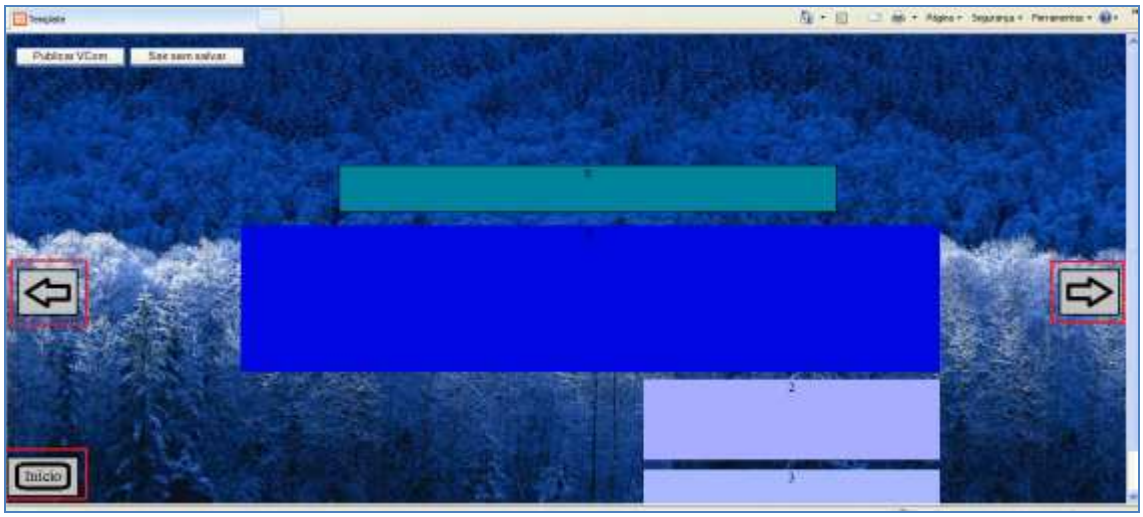


Figura 5.12: Navegação - Paginação

Na paginação as UPIs ficam todas na mesma página semelhantemente ao tipo de navegação “Estrutura Sequencial”, mas permanecem e são mostradas dependendo da ação do usuário (avançar ou voltar container). A diferença entre esses tipos de navegação está na interação com o template, quando se dá uma pseudo-transição de páginas na navegação do tipo “Paginação”.

Estrutura de Menus: Com este tipo de navegação o usuário terá acesso direto aos containers, semelhantemente ao tipo de navegação “Árvore de Links”. A diferença está na interação do participante com o ambiente.

Figura 5.13: Navegação – Menus

Como pode ser observado na Figura 5.13, o usuário pode escolher qualquer container e acessá-lo diretamente. Para exemplo, acessamos o 3º container (Tópico 2) e o template se comporta como na Figura 5.14. Como se pode observar ainda na Figura 5.14, o menu permanece no template, diferentemente da opção de navegação “Árvore de links”, na qual para ter acesso ao quadro de links, é necessário retornar ao início do template.

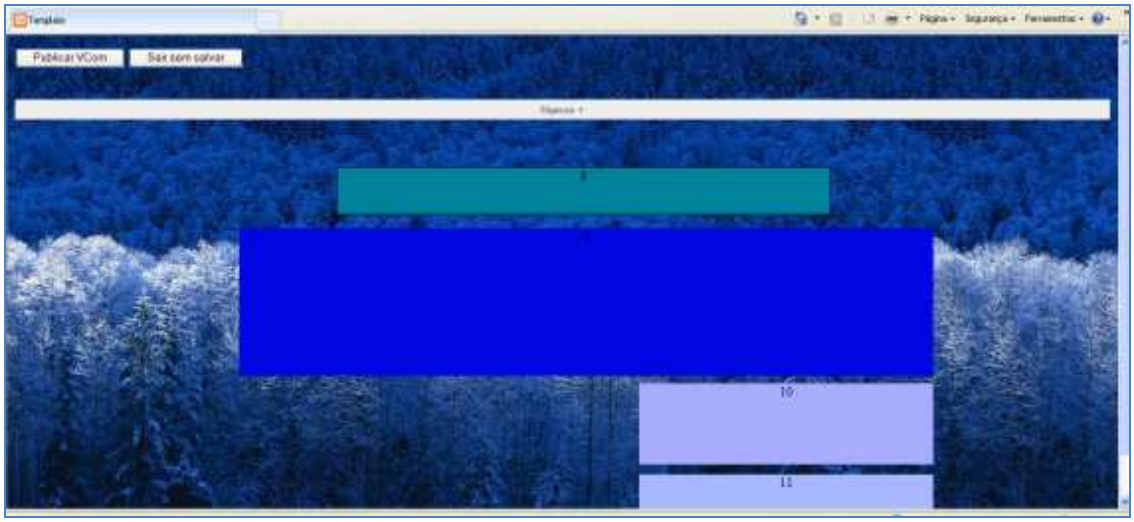


Figura 5.14: Menus - Container 3

Após escolher um dos quatro tipos de navegação existentes no Editor de Templates, o usuário poderá finalmente publicar o template de seu VCom por meio do botão “Publicar VCom” do módulo de navegação.

5.2.2 Editar Templates

A edição de templates funciona de forma similar ao processo de criação. Os passos da edição são os mesmos passos da criação. O usuário deverá obrigatoriamente utilizar todos os módulos na edição, mesmo que sua alteração seja uma específica de um único módulo. Isso é necessário devido a necessidade da ação do mecanismo de publicação (seção 5.2.4) que somente está disponível no final da utilização do editor.

Inicialmente, o usuário deve clicar no botão “Editar Template” da Figura 5.1. Esse botão dispara um evento que mostra a janela em destaque na Figura 5.15. O usuário deve escolher o template a ser editado e clicar em “Iniciar”.

Figura 5.15: Editar Templates

Continuaremos a demonstração do protótipo com o template “Template *Blog*”. Ao iniciar o processo de edição do template, o primeiro interpretador do Editor de Templates irá usar o arquivo XSL salvo na criação do template e remontar o *layout* conforme a Figura 5.5. O usuário fará suas alterações ou não e necessariamente deverá salvar as alterações.

No segundo passo da edição, o usuário poderá reutilizar o *design* previamente definido na criação do template clicando no botão “Usar *Design* Anterior” da Figura 5.6, ou então redefinir seu *design* a partir do princípio.

Finalmente, no terceiro e último passo, o participante do ambiente terá acesso à tela representada na Figura 5.8 e poderá escolher seu tipo de navegação preferido. Uma vez escolhida a navegação, o usuário terá a pré-visualização do template final (assim como na criação) e poderá publicar seu template.

5.2.3 Acessar Espaço Virtual

No projeto do Editor de Templates existe um espaço para cada usuário no qual são armazenadas as informações referentes ao template de seus espaços virtuais (*layout*, *design* e navegação). Essas informações são introduzidas nesse espaço após a publicação do template.

Assim que publicado, o template estará disponível para acesso direto do usuário pela opção “Meus Espaços Virtuais” na Figura 5.1. Uma vez clicado, esse botão dispara um evento que lança a janela destacada na Figura 5.16.

Figura 5.16: Acessar Espaço Virtual

Quando o espaço virtual é acessado diretamente, as informações do template (que já estão no espaço reservado ao usuário) se unirão à estrutura do VCom em seus elementos, os quais estarão aptos a receber as UPIs, como destacado na Figura 5.17.

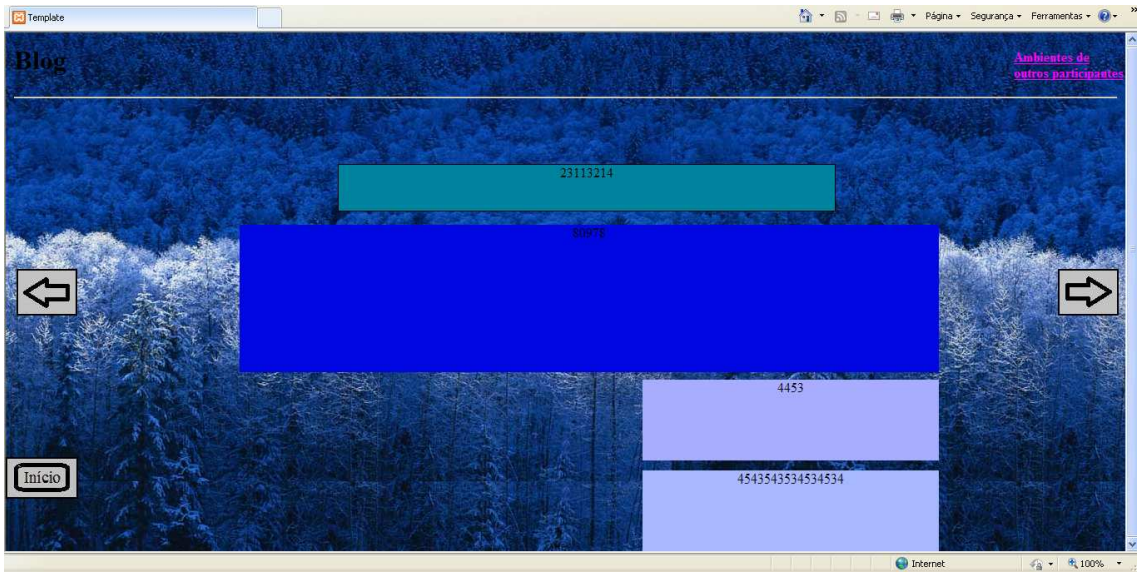


Figura 5.17: Espaço Virtual com template publicado

Na Figura 5.17, as UPIs são algarismos e não texto, como se pode notar e para este exemplo, definimos a navegação como do tipo “Paginação”. Na próxima seção descreveremos com mais detalhes o mecanismo de publicação.

5.2.4 Mecanismo de Publicação

Na publicação do template, automaticamente as definições de navegação serão documentadas em um arquivo XML e guardadas no módulo de navegação. Além desse fato, após a publicação do template, um novo arquivo

CSS é gerado documentando o *layout* da página para o acesso direto. Nesse caso, o documento XSL (que contém a definição de *layout*) permanece intacto em seu módulo caso o usuário queira redefinir o *layout* de seu template.

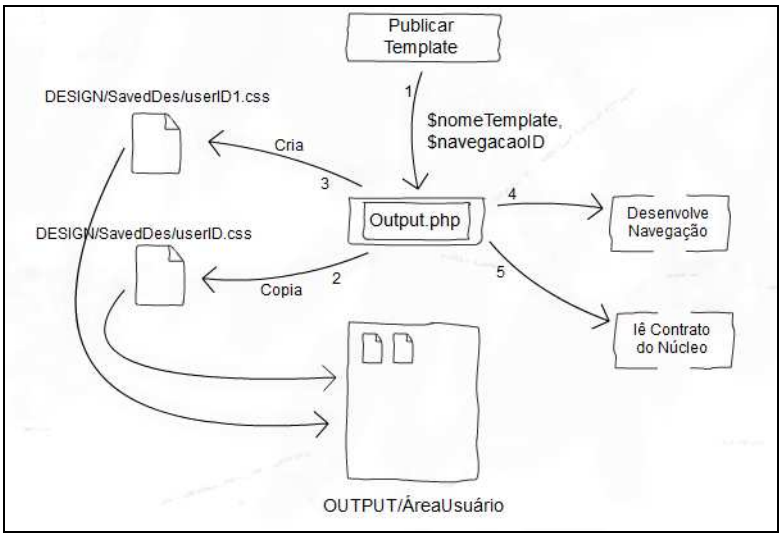


Figura 5.18: Mecanismo de Publicação

Na Figura 5.18 está representado os passos do ocorre quando se publica um template para um determinado VCom. Existe um módulo final, cuja principal fonte de programação é o documento Output.php. Esse programa recebe como entrada o nome do template e o tipo de navegação escolhida no módulo anterior, e então cria o documento CSS do *layout* e o copia junto com o CSS do *design* para o espaço do usuário.

A partir dos dados de entrada, o programa ainda monta a navegação e busca as UPIs no banco de dados a partir do contrato estabelecido com as regras do veículo de comunicação. A Seção 5.2.6 explica com mais exatidão o funcionamento da leitura do contrato do núcleo e o estabelecimento das visões.

5.2.5 Estrutura de Arquivos

Como o Editor de Templates foi separado em módulos, a sua estrutura de diretórios foi definida sobre esse princípio. Cada módulo é independente e possui seus próprios programas e regras de interface.

Como pode ser notado na Figura 5.19, as pastas “NAVIGATION”, “*LAYOUT*” e “*DESIGN*” contêm subpastas para armazenar os arquivos gerados e salvos

(Saved...) pelos módulos na passagem do usuário por eles. Além disso, os módulos apresentam documentos “.js” (javascript) e “.php” que efetuam a programação da página e documentos CSS que montam o estilo da interface de cada módulo.

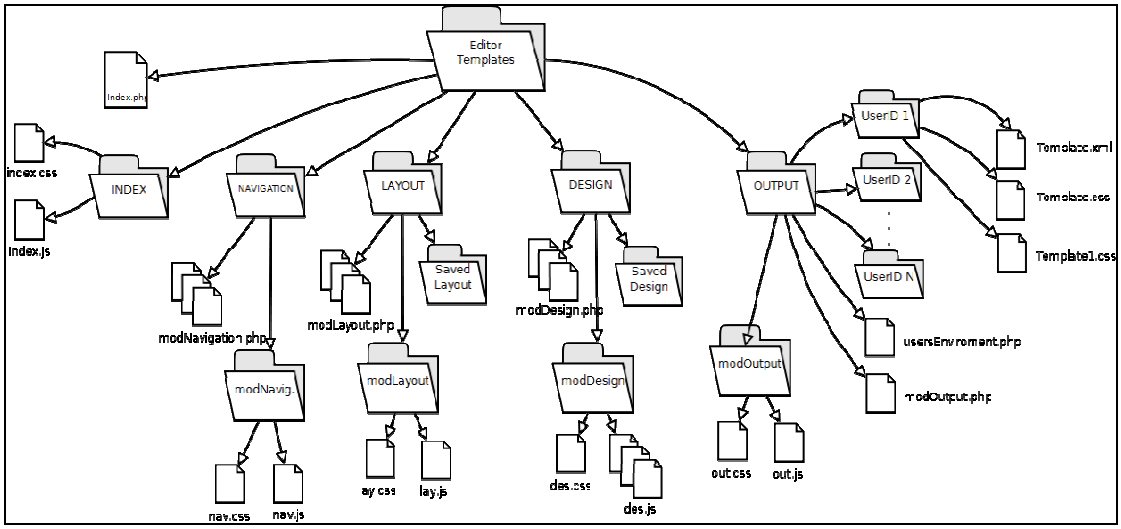


Figura 5.19: Estrutura de Diretórios do Editor de Templates

A pasta OUTPUT armazena o quarto módulo do editor e é a partir dela que o mecanismo de publicação é aplicado. Essa pasta, especialmente é destacada, pois guarda o espaço de cada usuário com documentos que são utilizados pelo programa “modOutput.php” para montar o espaço virtual do participante. Esses documentos são nomeados como “Template.xml” que contêm informações sobre navegação, “Template.css” e “Template1.css” que montam o *design* e *layout*, respectivamente, do espaço virtual que é acessado diretamente.

5.2.6 Mecanismo de Visões

Inicialmente, as permissões de leitura e escrita são dadas no desenvolvimento do veículo de comunicação. Na publicação do VCom essas informações são interpretadas e inseridas em tabelas do banco de dados do MOrFEu. Essas tabelas possuem o identificador do usuário e do VCom, o container e elemento da interface e as data de inicio e fim da edição/visualização da UPI. Essas informações permitem o estabelecimento de um contrato de visões.

O objetivo do mecanismo de visões é funcionar como um interpretador do contrato, inserindo as UPIs que estão no banco de dados nos elementos de interface desenhados pelo usuário.

Vamos exemplificar o funcionamento desse mecanismo usando a figura de dois participantes, sendo que o participante 1 definiu seu template, como na Figura 5.17, e o participante 2 definiu seu template como na Figura 5.20.

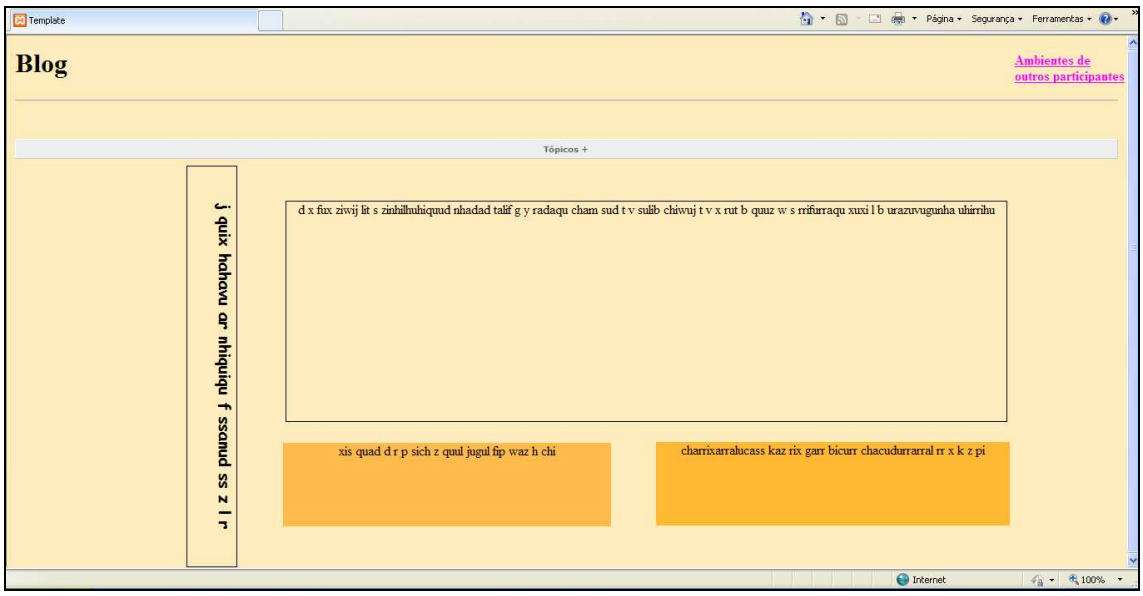


Figura 5.20: Template do Participante 2

Ao clicar no link “Ambiente de outros participantes” na Figura 5.20, será aberta uma janela como desenhada na Figura 5.21. Como existem somente 2 participantes nesse ambiente, o participante 2 terá acesso apenas às UPIs do participante 1, como em destaque na imagem.

Figura 5.21: Tela de acesso ao espaço de outros usuários do ambiente

O participante 2, ao selecionar o participante 1, poderá visualizar o conteúdo do *Blog* oposto de acordo com seu template. Repare na Figura 5.22 que o participante 2, está vendo o conteúdo do *Blog* do participante 1 (conferido na Figura 5.17) em seu template. Ou seja, o participante 2 tem uma visão inteiramente particular do *Blog* do participante 1, e vice-versa.

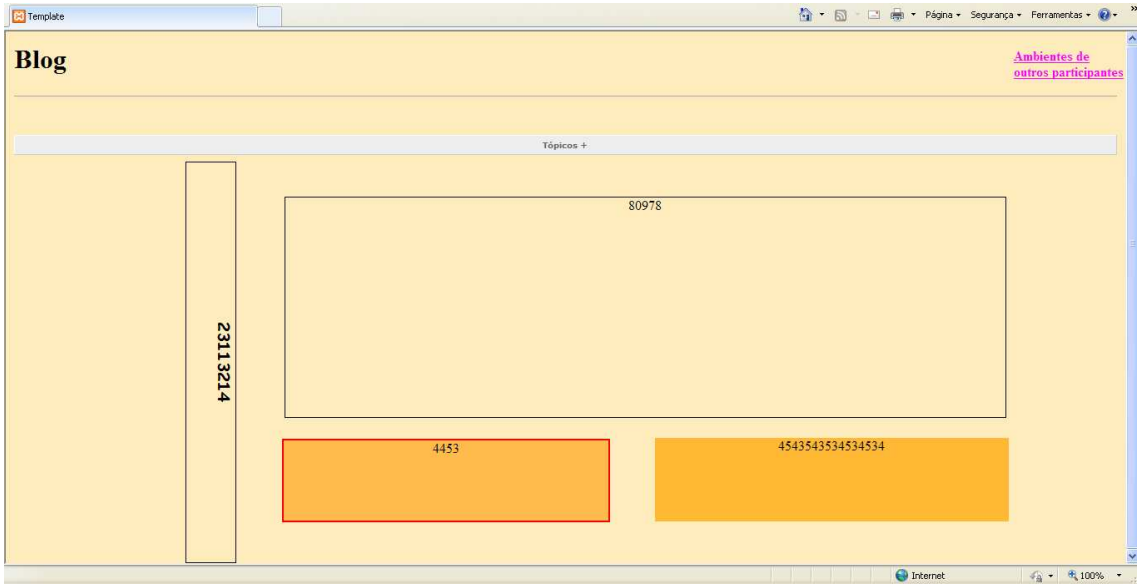


Figura 5.22: Visão do participante 2

Repare ainda na imagem que a caixa de comentário da esquerda está com a borda vermelha. Isto ocorre, pois no contrato estabelecido, esse elemento pode ser editado pelo participante que o está acessando.

Figura 5.23: Editor de UPIs

Ao colocar o ponteiro do mouse sobre o elemento, a borda aparece, e o clique sobre ele dispara um evento que chama o editor de UPIs, mostrado na Figura

5.23. Ou seja, a visão também estabelece a possibilidade de edição de conteúdo conforme a interpretação do contrato.

A figura do contrato é totalmente abstrata. Existem duas tabelas principais no banco de dados do MOrFEu que estabelecem as UPIs que serão mostradas nos elementos específicos, e os usuários que deverão ter o acesso sobre eles. O contrato nada mais é do que uma consulta nessas tabelas, feita por meio de codificação no documento “Output.php”.

5.3 LIMITAÇÕES DA IMPLEMENTAÇÃO DO EDITOR DE TEMPLATES

A primeira limitação do protótipo, é que atualmente funciona apenas para o navegador Internet Explorer. É extremamente dispendioso projetar arquiteturas que funcionem em todos os navegadores, então consideramos priorizar um dos navegadores mais utilizado na atualidade.

Outra limitação dessa primeira versão da implementação do editor de templates está no módulo de *design*, o qual não apresenta as últimas alterações do usuário, sendo necessário redefinir a partir do princípio as alterações, mesmo que seja uma mudança simples como tamanho de fonte de um elemento, por exemplo.

Com relação às visões, os participantes quando acessam o ambiente de outros participantes, vêm o conteúdo de acordo com o template estabelecido por eles próprios. As visões se propõem a dar ao usuário um acesso inteiramente particular. Contudo, o protótipo não permite ao participante 1 ver o ambiente do participante 2 com o template do participante 2, apesar da proposta do Editor de Templates permitir que isso ocorra.

5.4 CONCLUSÃO

O desenvolvimento de interfaces pode ser um processo agradável e pode facilitar muito o acesso e a interação do participante com o ambiente. Quando a

interface é flexível, isto é, passível de customização, as necessidades do usuário podem ser supridas sem a necessidade de intervenção de segundos ou terceiros. A primeira versão do Editor de Templates tem o propósito geral de tornar o usuário dono da interface de seu ambiente, estabelecendo para ele um template, que é a descrição de sua aparência.

Percebemos que uma arquitetura modular oferece uma distribuição de responsabilidades no desenho da aparência e permite uma edição independente para as partes que compõem a interface. Além disso, esse tipo de arquitetura facilita o processo de manutenção e atualização do sistema, uma vez que cada módulo possui suas próprias diretivas de programação.

Por fim, neste capítulo apresentamos as tecnologias usadas no processo do desenvolvimento do protótipo, os módulos que o compõem, sua estrutura de arquivos e diretórios, as limitações e os mecanismos atuantes sobre a publicação da interface e de interpretação do contrato de visões sobre o ambiente.

CAPÍTULO 6 EXEMPLOS DE USO

Para avaliar a proposta elaborada neste trabalho, usaremos a arquitetura “Debate de Teses” como principal exemplo de uso e então analisaremos os aspectos do suporte computacional oferecido pelo Editor de Templates.

Este capítulo está dividido da seguinte forma: Na Seção 6.1 descreveremos a arquitetura pedagógica, suas etapas e o fluxo de atividades. Na Seção 6.2 apresentaremos a utilização do debate de teses no protótipo, incluindo os arquivos que são gerados durante sua utilização, a Seção 6.3 apresenta outro exemplo de uso, dessa vez com um ambiente incomum e finalmente, na Seção 6.4, as conclusões sobre o capítulo.

6.1 ARQUITETURA PEDAGÓGICA DEBATE DE TESES

Segundo [NEVADO, 2011], a construção de conhecimento sobre um determinado assunto requer que o sujeito, partindo de seu conhecimento prévio, provocado por situações de desequilíbrio, faça reconstruções progressivas de forma que suas estruturas conceituais consigam assimilar novas situações.

Retirando em parte a responsabilidade do professor de ser o único abonador da construção do conhecimento do aluno, e baseada em princípios construtivistas, a arquitetura “Debate de Teses” busca possibilidades de enriquecimento do pensamento individual e coletivo e também a produção de novidades resultantes das trocas cooperativas.

6.1.1 Debatendo Teses

A arquitetura é projetada para funcionar sobre a plataforma web, de forma assíncrona no que diz respeito aos participantes do debate trabalhando em tempos distintos, e de forma síncrona, mas com os participantes somente

tendo acesso ao espaço de outros participantes (e à parte de seu próprio espaço) no prazo e data determinados pelo coordenador do debate. De acordo com [NEVADO, 2011], buscando uma sistematização das participações, aliada a uma preocupação efetiva da contribuição de cada sujeito para o processo, foram definidos os seguintes elementos estruturantes:

i) Inicialmente, algumas afirmações são levantadas pelo professor (ou alunos) nas quais são apresentadas concepções sobre um assunto de interesse comum e que geralmente têm potencial para gerar grandes desequilíbrios.

ii) Cada participante deve manifestar-se com respeito a cada tese, indicando se concorda, discorda ou se não sabe decidir. Em qualquer um dos casos deve escrever uma justificativa baseada em argumentos e evidências.

iii) A argumentação inicial é revisada em um terceiro momento por outros dois participantes, não para confrontar ideias, mas para validar a consistência dos argumentos frente às evidências apresentadas.

iv) No quarto momento, o argumentador poderá aceitar ou não a revisão. No caso de rejeição, o argumentador poderá fazer uma réplica. Caso contrário, ele estaria reforçando ou reconsiderando sua posição.

v) Além da elaboração das teses, o professor se concentra na observação das manifestações do participante, apoiando nos aspectos metodológicos da construção de argumentações, revisões e posicionamentos.

Após o término do debate todos os participantes têm acesso, para leitura, aos espaços individuais dos demais participantes. Eventualmente, com a permissão dos diversos participantes, o debate completo é tornado público.[4]

O debate de teses é uma arquitetura bastante complexa em seu fluxo de atividades e coordenação dos participantes do espaço virtual. Além disso, apresenta uma estrutura de interação que é muito mais complexa que a maioria dos veículos de comunicação conhecidos, no que diz respeito às restrições temporais e recursos de customização. Por esse motivo entendemos

[4] Uma versão preliminar do ambiente está correntemente implementada em PHP e em breve será disponibilizada para distribuição em sw livre. http://lied.inf.ufes.br/acesso2/debateteses/dt/.

que se o processo de criação de templates para essa arquitetura for satisfatório, também será para outras famílias de VComs.

6.1.2 Etapas do Processo de Debate

A seguir, descreveremos como acontecerão as interações entre os participantes do ambiente e fluxo de trabalho no processo. As atividades no debate serão realizadas durante a execução de 5 passos que estarão disponíveis para edição dependendo do prazo elaborado pelo coordenador.

(i) Apresentação da tese;
(ii) Posicionamento Inicial e argumentação;
(iii) Revisão sobre a argumentação;
(iv) Réplica sobre a revisão;
(v) Posicionamento Final;

Inicialmente, o coordenador do ambiente, apresentará as teses a serem discutidas, e então (na segunda etapa) cada participante do ambiente firmará um posicionamento inicial dizendo se concorda, discorda ou não sabe decidir, e deverá apresentar também uma argumentação sobre esse posicionamento.

Na terceira etapa as revisões sobre a argumentação serão elaboradas. Os indivíduos então serão delegados à tarefa de acessarem o espaço de outros participantes. A delegação poderá ser aleatória ou direta, dependendo da determinação do coordenador. Nessa etapa nenhuma atividade de edição poderá ser realizada no ambiente do próprio participante.

Uma réplica poderá ser escrita na quarta etapa, caso o argumentador não concorde com a revisão que recebeu em seu argumento. Caso contrário, o espaço de edição reservado para essa etapa poderá ficar em branco, o que torna a realização dessa etapa, opcional.

Finalmente, um posicionamento final será elaborado pelo argumentador. Esta última etapa constitui a conclusão sobre o debate, na qual os participantes poderão descrever suas experiências e estabelecer sua opinião.

A Figura 6.1, descreve os passos do fluxo de trabalho dos participantes no Debate de Teses. Inicialmente o Administrador cria um debate e delega funções (ou papéis) aos usuários, o que os torna participantes do ambiente.

O Mediador deve criar o cronograma (prazos para edição/leitura de produções no ambiente) e escrever as teses para serem debatidas. O mediador a qualquer momento pode realizar comentários nas etapas do debate. O argumentador então apresenta seu argumento para que posteriormente o revisor o revise. A réplica sobre a revisão é alternativa, como pode ser visto no diagrama. No último passo do fluxo, o argumentador escreve sua posição final.

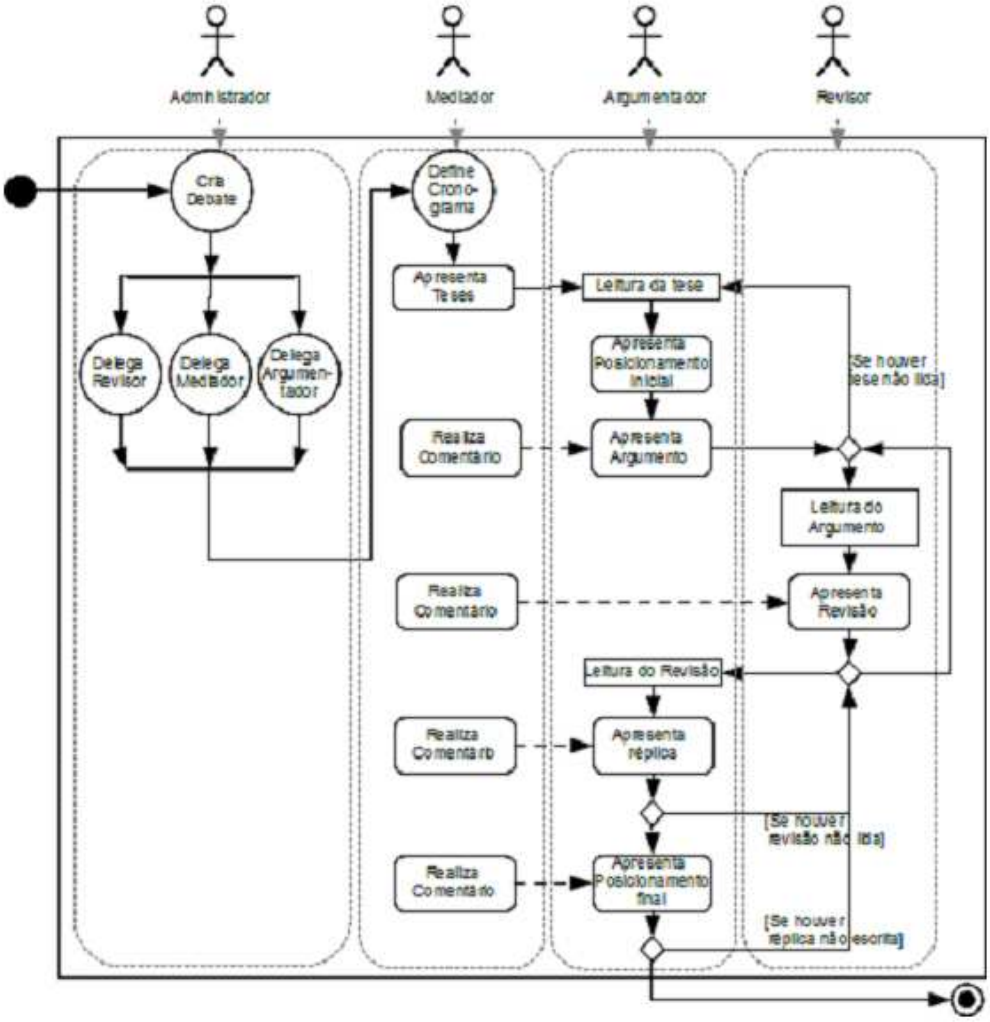


Figura 6.1: Atividades no Debate de Teses [VIEIRA, 2011]

Um mesmo usuário não pode desempenhar o papel de Argumentador e Revisor ao mesmo tempo, mas em diferentes momentos pode, ou seja, o participante que foi argumentador antes pode ser revisor na etapa seguinte.

6.2 APLICAÇÃO DE TEMPLATES NO AMBIENTE “DEBATE DE TESES”

Neste exemplo de uso, definiremos usuários denominados participante 1, participante 2 e participante 3, sendo que o segundo e terceiro desempenharão os papeis de revisor e argumentador e o primeiro, de mediador.

```
<?xml version="1.0"?>
<vcom id=1>
    <header>
        <author>Jean Baptiste Debret</author>
        <created>20120114</created>
        <VComName>Debate de Teses</VComName>
        <class>Discussao</class>
    </header>
    <title>Debate de Teses</title>
    <amount>10</amount>
    <container>
        <T>
            <term>2</term>
            <awnsers>1</awnsers>
            <users>
                ...
            </users>
            <name>Tese</name>
        </T>
        <PI>
            <term>2</term>
            <awnsers>1</awnsers>
            <users>
                ...
            </users>
            <name>Posicionamento Inicial</name>
        </PI>
        <R>
            <term>3</term>
            <awnsers>2</awnsers>
            <users>
                ...
            </users>
            <name>Revisao</name>
        </R>
        <REP>
            <term>3</term>
            <awnsers>2</awnsers>
            <users>
                ...
            </users>
            <name>Replica</name>
        </REP>
        <PF>
            <term>2</term>
            <awnsers>1</awnsers>
            <users>
                ...
            </users>
            <name>Posicionamento Final</name>
        </PF>
    <container>
</vcom>
```

Figura 6.2: VCom em XML: Debate de Teses

Com o intuito de uma visualização geral, vamos abranger o exemplo de uso desde a definição do VCom até a construção do template. Podemos visualizar uma descrição do VCom no documento XML mostrado na Figura 6.2.

Como se pode notar, esse documento XML apresenta uma modelo de definição bem simples do veículo de comunicação. Nele existe um cabeçalho, que é útil para busca de VComs na biblioteca, apresenta um Título e um campo para definir a quantidade de containers (*amount*) que nesse caso é 10.

Para este exemplo, definimos que todos os elementos da página, exceto o título, sofreriam navegação. Então, para simplificar o documento usamos apenas um container (que sofrerá navegação) e deixamos o título (fixo) fora dele.

6.2.1 Suporte à Descrição de Interface Visual

Baseado no modelo de descrição de VCom exposto na Figura 6.2, foi definido um template para cada participante. A Figura 6.3[5] mostra o template final, definido pelo participante 1 e a Figura 6.4, o template do participante 2.

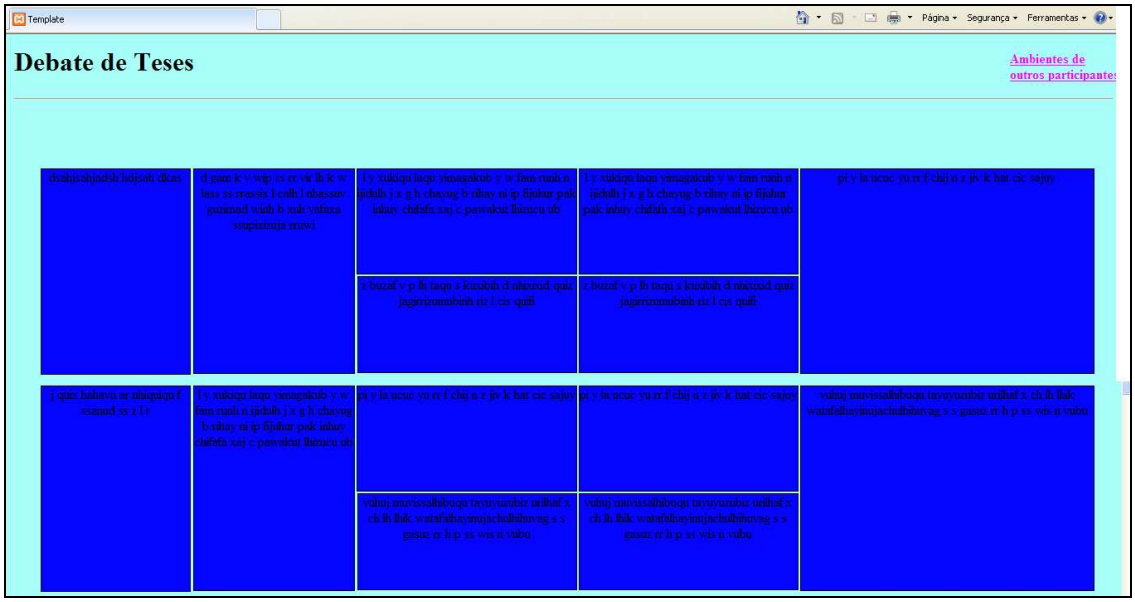


Figura 6.3: Template do Participante 1

[5] O texto presente na figura é meramente ilustrativo.

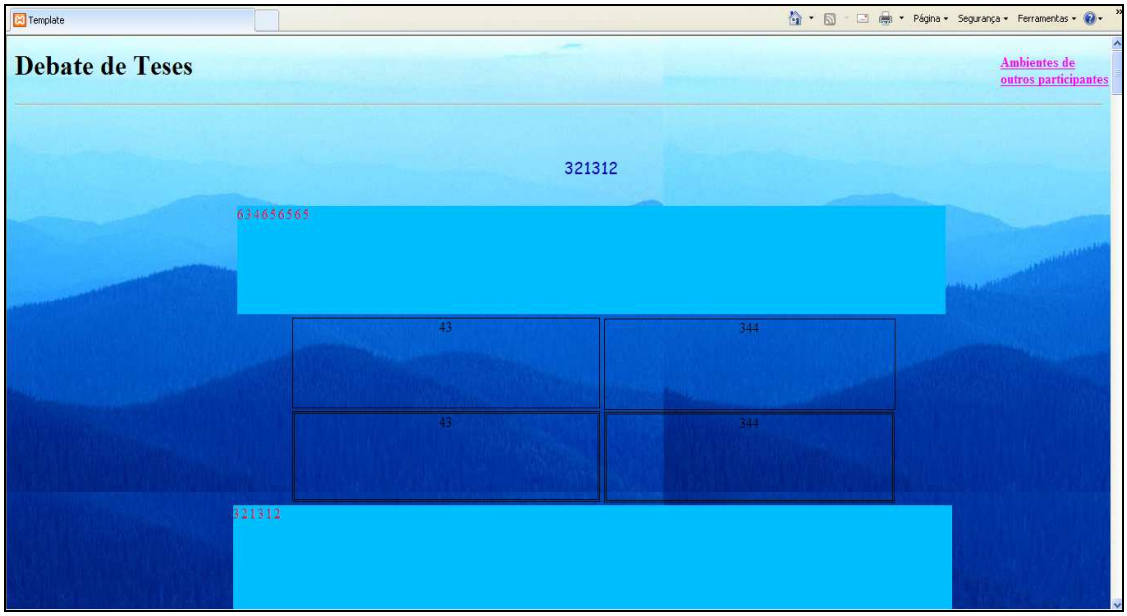


Figura 6.4: Template do Participante 2

Durante o processo de criação do template, foram gerados os documentos que contêm as definições de interface, estabelecidas no ato de salvar o template, no decorrer da utilização de cada módulo. Os documentos que mostraremos a seguir, são referentes ao template desenvolvido pelo participante 2.

```
13      <xsl:for-each select="vcom/componente">
14          <div id="T" class="drsElement" style="left:42px; top:62px; width:181px; height:249px;" >
15              <xsl:value-of select="T/name"/>
16          </div>
17          <div id="PI" class="drsElement" style="left:226px; top:62px; width:195px; height:248px;" >
18              <xsl:value-of select="PI/name"/>
19          </div>
20          <div id="R1" class="drsElement" style="left:423px; top:62px; width:266px; height:128px;" >
21              <xsl:value-of select="R/name"/>
22          </div>
23          <div id="R2" class="drsElement" style="left:424px; top:192px; width:265px; height:117px;" >
24              <xsl:value-of select="R/name"/>
25          </div>
26          <div id="REP1" class="drsElement" style="left:691px; top:62px; width:265px; height:128px;" >
27              <xsl:value-of select="REP/name"/>
28          </div>
29          <div id="REP2" class="drsElement" style="left:691px; top:192px; width:265px; height:117px;" >
30              <xsl:value-of select="REP/name"/>
31          </div>
32          <div id="PF" class="drsElement" style="left:958px; top:62px; width:355px; height:248px;" >
33              <xsl:value-of select="PF/name"/>
34          </div>
35      </xsl:for-each>
```

Figura 6.5: Fragmento do documento XSL que documenta o *layout* do Template

O primeiro documento que obtemos é o XSL que contém as definições de *layout* do template. A Figura 6.5 mostra um fragmento desse documento. Nele estão definidas as posições absolutas dos elementos na tela, além dos objetos de programação necessários para a transformação do XML em HTML, ou seja, da transformação da documentação escrita do VCom em interface gráfica.

```
23          border-color : #0C0004;
24      }
25      .R2{
26          background-color : #0404FF;
27          border-width : 1px;
28          border-style : solid;
29          border-color : #0C0004;
30      }
31      .PI{
32          background-color : #0404FF;
33          border-width : 1px;
34          border-style : solid;
35          border-color : #0C0004;
36      }
37      .PF{
38          background-color : #0404FF;
39          border-width : 1px;
40          border-style : solid;
41          border-color : #0C0004;
42      }
43      body {
44          background-color : #A8FFFA;
45          text-align : center;
46      }
47
```

Figura 6.6: Documento CSS com as definições de *design* do Template

O documento CSS obtido como segundo elemento na utilização do Editor de Templates tem um fragmento representado na Figura 6.6. Todos os elementos possuem um identificador (id), que é definido com o número do container concatenado ao nome do elemento. A partir dessa identificação, cada elemento terá sua própria definição de *design*.

O editor de Templates é capaz de satisfazer os requisitos de interface do Debate de Teses, de forma flexível. Podemos dizer que o editor se mostrou capaz de permitir a determinação de uma apresentação visual pelas mãos do próprio participante. Porém, muito mais complexo que satisfazer os requisitos de interface, é satisfazer os requisitos funcionais de atividade no ambiente, que inclui as permissões de acesso aos espaços dos participantes, as ações que podem ser realizadas e os prazos para que elas ocorram. Esses requisitos serão mais bem discutidos a seguir.

6.2.2 Suporte a Diferentes Visões do Ambiente

Os participantes do ambiente necessariamente precisam ser encaixados em grupos, ou perfis, a fim de estabelecer parâmetros de acesso que envolvam edição e visualização de conteúdo em um espaço de tempo. A seguir demonstraremos como essas restrições deverão ser aplicadas pelo editor de templates na apresentação do espaço virtual e de seu conteúdo.

Restrições de Visualização:

As restrições de visualização são aplicadas a todos os participantes do ambiente, com exceção do participante 3 (mediador) que tem acesso de leitura no espaço dos demais participantes do ambiente.

Com a existência de um quarto participante P4, que teria os mesmo privilégios dos participantes P1 e P2, o mediador poderia definir uma restrição cíclica, na qual o P1 poderia visualizar o conteúdo do espaço de P2, mas não o de P4. P2 teria acesso ao espaço de P4, mas não ao de P1, e P4 acessaria P1, mas não poderia acessar P2.

As restrições de visualização são definidas na construção do VCom, geralmente pelo usuário Mediador, e devem além de respeitar a integridade de usuários, respeitar a integridade de prazo.

Restrições de Edição:

As restrições de edição são aplicadas para que a interação entre os participantes possa acontecer. As interações englobam principalmente os atos de fazer revisão sobre argumentação e fazer réplica sobre revisão.

Neste exemplo de uso, o usuário Mediador não tem permissão de escrita (edição) sobre os elementos dos demais participantes, apesar de poder visualizar o espaço de todos os participantes.

Assim como as restrições de visualização, as restrições de edição, são definidas na construção do VCom, geralmente pelo usuário Mediador, e apesar de pré-definidas, necessariamente entrarão em vigor, apenas durante prazo estabelecido.

Restrições Temporais:

No momento da criação do VCom, é armazenado em sua descrição, a data de criação. Criamos o Debate de Teses para este exemplo de uso no dia 22 de

janeiro e definimos a duração de 5 dias para o debate. Os prazos de interação são os seguintes: 1 dia para leitura da tese e escrita do posicionamento inicial. 1 dia para efetuar a revisão do posicionamento inicial de outro participante. 1 dia para réplica, e dias para efetuar o posicionamento final.

A partir desse prazo, mostraremos como ficaria a visão do participante 1 com o decorrer do passar dos dias. Inicialmente, dia 22 de janeiro ele lê a tese e escreve seu Posicionamento Inicial (PI) como mostrado na Figura 6.7. Repare que a borda do elemento PI no primeiro container está avermelhada. Isso significa que o ponteiro do mouse do participante está sobre o elemento e seu conteúdo pode ser editado.

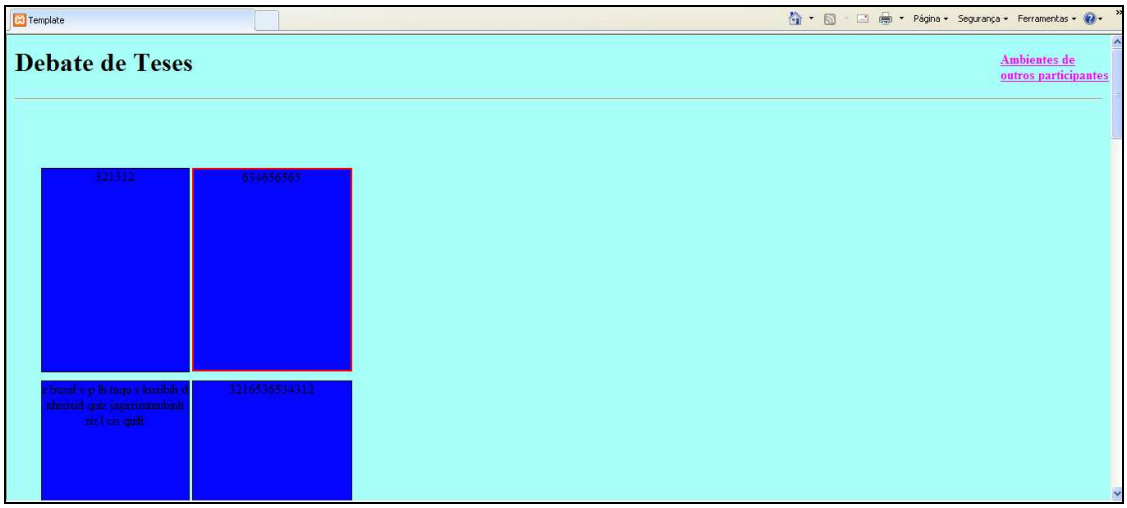


Figura 6.7: Dia 22 de Janeiro, espaço do participante 1

No dia 23, o acesso ao espaço de outro participante (determinado pelo mediador) estará disponível. Para isso, o usuário deverá clicar no *link* “Ambiente de outros Participantes” no canto superior direito da Figura 6.7. No momento que esse acesso é feito, o participante P1 verá o espaço do participante P2 como na Figura 6.8.

Figura 6.8: Dia 23 - Conteúdo de P2 acessado por P1

Como se pode observar na imagem, P1 passa a ter acesso ao posicionamento inicial de P2 e pode escrever sua revisão. Como pode ser visto ainda, P1 não tem acesso nem de leitura nem de escrita ao espaço para a segunda revisão que deverá ser feita por um terceiro participante.

No dia 24, P1, assim como os demais participantes, já terão suas revisões efetuadas, e poderão postar suas réplicas, como pode ser observado na Figura 6.9. A partir desse ponto, o usuário P1 não poderá mais efetuar edição sobre seu conteúdo no espaço de P2, apesar de permanecer visualizando-o.

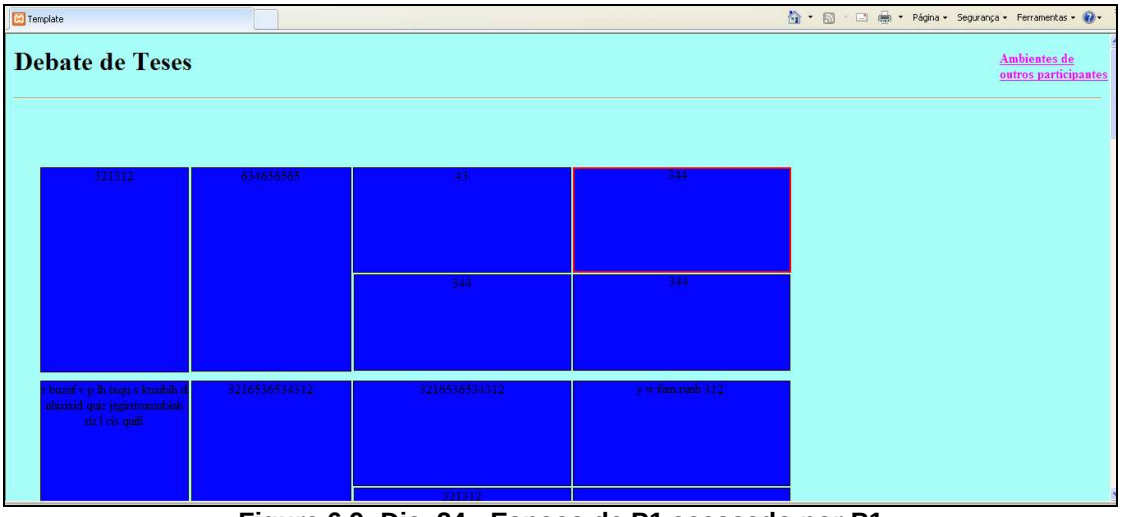


Figura 6.9: Dia 24 - Espaço de P1 acessado por P1

Após a postagem da réplica os participantes finalmente poderão escrever seu posicionamento final (PF) sobre tese, a partir do dia 25. Podemos ver na Figura 6.10 que o participante P1 está prestes a editar o elemento PF do container 1.

Figura 6.10: Dia 25 - P1 escrevendo posicionamento final em seu espaço

No final do debate, a partir do dia 26, os participantes poderão ter total acesso de visualização dos ambientes entre si, se assim o mediador determinar. Na Figura 6.3, no início da seção anterior, temos um desenho da visão do participante P1 sobre o conteúdo do espaço de P2 no final do debate.

6.3 APLICAÇÃO DE TEMPLATES NO AMBIENTE “CONFRONTO DE OPINIÕES”

Na seção 3.1.3 citamos um ambiente de discussão de ideias e dissemos que atualmente não existem ferramentas na web que possam fornecer suporte para sua aplicação. Demos a esse ambiente o nome de “Confronto de Opiniões” ou simplesmente “Confronto”.

```
<VCOM>
    <HEADER>
        ...
    <HEADER/>
    <VCOMSTRUCT>
        <CONTAINER Navigation="False">
                <TEXT>
                    <TESE>
                    No atual âmbito das salas de aula, o computador é a melhor
                    ferramenta que um professor pode ter em mãos para facilitar a
                    construção do processo de ensino-aprendizagem. Quadros, giz e
                    livros precisam abrir espaço para projetores, mouses e tablets.
                    </TESE>
                </TEXT>
        </CONTAINER>
        <CONTAINER Navigation="False">
                <IMAGE>
                    EdTempates/Images/x.jpg
                </IMAGE>
        </CONTAINER>
        <AMOUNT>5<AMOUNT>
        <CONTAINER Navigation="True">
            <AC>
                <TERM>2</TERM>
                <AWNSERS>1</AWNSERS>
                <NAME>Argumento Contra</NAME>
            </AC>
            <AF>
                <TERM>2</TERM>
                <AWNSERS>1</AWNSERS>
                <NAME>Argumento Favor</NAME>
            </AF>
            <PRC>
                <TERM>3</TERM>
                <AWNSERS>2</AWNSERS>
                <NAME>Prova C.</NAME>
            </PRC>
            <PRF>
                <TERM>3</TERM>
                <AWNSERS>2</AWNSERS>
                <NAME>Prova F.</NAME>
            </PRF>
        </CONTAINER>
    </VCOMSTRUCT>
<VCOM/>
```

Figura 6.11: Descrição do VCom "Confronto"

Primeiro representamos o ambiente na linguagem de descrição de VComs que desenvolvemos para este trabalho. A Figura 6.11 mostra o documento elaborado, pronto para ser utilizado na etapa de criação do template. Repare que este confronto terá cinco (AMOUT) argumentos a favor e contra a tese que está dentro de um container que é fixo (Navigation = False). Para cada tese apresentada, o grupo terá que apresentar duas provas que sustente sua argumentação. Essas provas podem ser citações de referência de trabalhos na área, por exemplo.

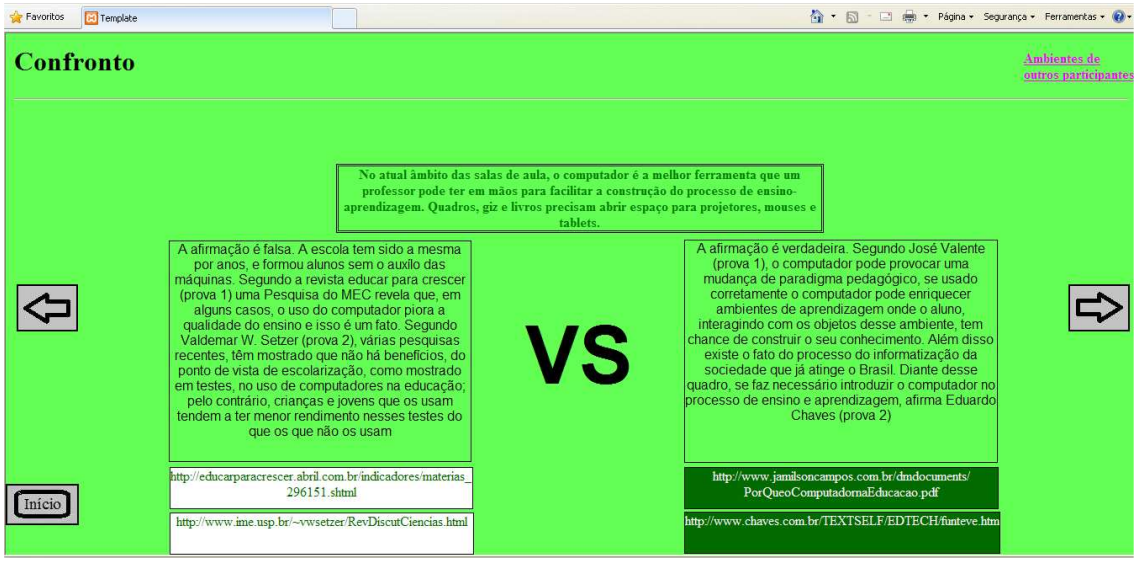


Figura 6.12: Template elaborado pelo Grupo 1 para o Confronto

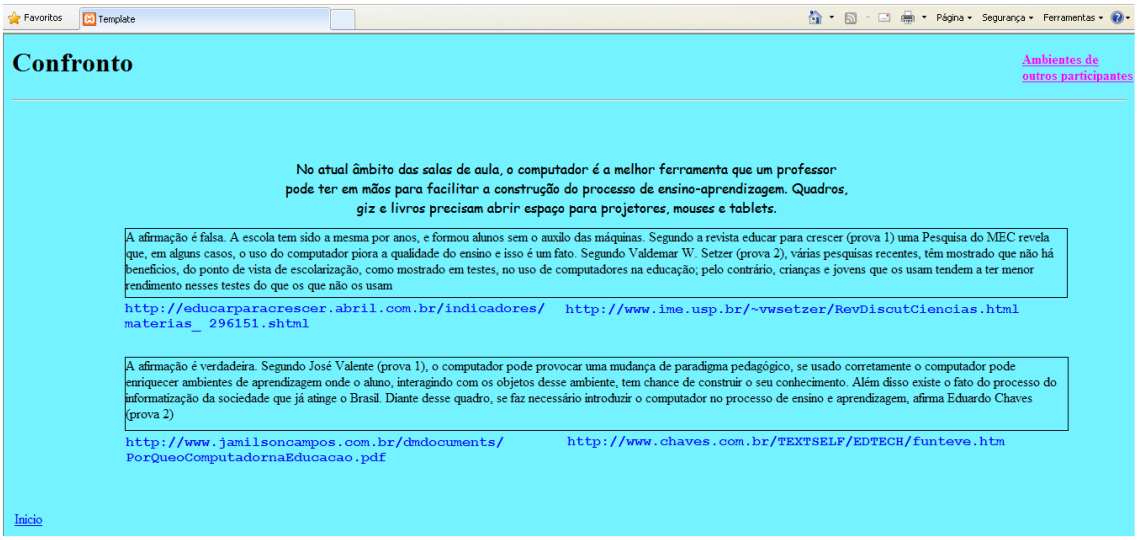


Figura 6.13: Template elaborado pelo Grupo 2 para o Confronto

Nas Figuras 6.12 e 6.13 estão os templates elaborados para o ambiente pelos grupos a favor e contra ao que está escrito na tese. Repare que o template do Grupo 1 apresenta o símbolo “VS” que, na verdade, é uma imagem (descrita na linguagem pela marcação <IMAGE>). Já o Grupo 2 optou por não utilizar o símbolo, deixando-o escondido na página. Essa opção é dada na utilização do módulo de *Design* do editor.

Nas figuras estão descritas apenas os primeiros argumentos e provas dos cinco que ainda serão realizados, porém, foi determinado que há um prazo para cada argumentação ser realizada que é indicado por um início e um fim. Nesse exemplo o início da segunda fase de argumentações deverá acontecer

no próximo dia.Agora as segundas argumentações podem confrontar as primeiras ao invés de confrontar a tese.

Ao final do prazo da apresentação dos argumentos prós e contras, o mediador pode fazer uma análise dos argumentos de cada grupo, indicando ou não vencedores. Para isso bastaria uma pequena edição no VCom para adição de um local na página que receba este posicionamento do mediador.

6.4 CONCLUSÃO

O Suporte computacional oferecido para a proposta do Editor de Templates contempla os requisitos estáticos e dinâmicos nas interações do participante com o ambiente por meio de sua interface gráfica. Neste capítulo mostramos como se dá o fluxo de trabalho por meio da utilização dos templates em um ambiente virtual colaborativo de complexidade considerável.

Concluímos que os requisitos para criação de apresentação de espaço virtual, restrições de temporalidade e especificação de acessos dos participantes foram atendidos adequadamente com as soluções oferecidas pelo template.

CAPÍTULO 7 CONSIDERAÇÕES FINAIS

Flexibilidade nos ambientes virtuais existentes é desejável em larga escala, contudo os ambientes atuais, centrados em ferramentas, não estão aptos a serem moldados de acordo com a necessidade ou preferência de seus utilizadores. Esses ambientes foram amplamente estudados e avaliados, inclusive empiricamente, o que viabilizou a base para concepção do MOrFEu [MENEZES et al, 2008].

A ferramenta MOrFEu, permite a descrição e elaboração de ambientes virtuais colaborativos por meio de seus componentes: o núcleo [VIEIRA, 2011] que viabiliza e impõe os limites para a descrição de atividades, o Editor de VCom [RANGEL, 2011], que descreve a estrutura do ambiente, seus participantes e os acessos de cada um deles, e agora o Editor de Templates, que promove a descrição da apresentação do espaço virtual e interpreta as restrições de acessos dos usuário por meio do mecanismos de visões.

Durante todo o estudo realizado neste trabalho, percebemos que os ambientes colaborativos atuais, para ensino-aprendizagem principalmente, não suportam o processo de autoria na camada de apresentação dos espaços virtuais. Esse fato promove a necessidade de uma ferramenta flexível que permita a descrição de interface gráfica para um melhor atendimento das necessidades ou preferências dos participantes do ambiente.

Ainda percebemos que para ambientes com finalidade produção cooperativa e gerência de conhecimento ter bom aproveitamento por parte de seus participantes, é necessário uma interface apropriada, baseada em princípios ergonômicos, fazendo com que as ferramentas, além de cumprirem sua função específica, com eficiência e eficácia, atraiam por sua beleza e/ou seu estilo.

Ao fazer uso dos conceitos de manipulação direta e interfaces exploráveis, dotamos o usuário de poder para executar suas ações e ao mesmo tempo tornar a experiência prazerosa, deixando-o satisfeito [ORLANDO, 2009]. Alcançamos, então, o propósito de obter interfaces adequadas, associadas ao conceito de usabilidade: produzir interfaces fáceis de usar, de aprender, de memorizar, que permitem uma experiência agradável e produtiva, na qual ansiedade e erros sejam diminuídos.

Este trabalho propôs então, uma ferramenta para introduzir na camada de apresentação dos ambientes virtuais, flexibilidade e customização de interface gráfica e um mecanismo que oferece uma visão particular de outros ambientes, mesmo que eles estejam customizados de outra forma.

Nossa contribuição trata-se de um editor que facilita o desenvolvimento e a manutenção de interfaces para espaços virtuais, dando ênfase na separação entre a apresentação, conteúdo e a estrutura do ambiente, sem que um elemento interfira no outro. Além disso, nossa proposta visa colocar o indivíduo frente às suas percepções dando-lhe a possibilidade de descobrir como tornar sua interação com o ambiente mais interessante ou mais agradável, mediante uma interface construída a seu sabor.

A abordagem apresentada considerou os ambientes virtuais convencionais e ainda apresenta algumas limitações que tornam o modelo de templates ainda incompleto, mas que permite a possibilidade de evolução. Um exemplo disso está no módulo de navegação. Existem quatro maneiras de navegação apenas. Nesse caso, o usuário escolhe sua navegação, e não a cria como nos outros módulos, quando o propósito principal do MOrFEu é a flexibilidade dos ambientes.

Com este trabalho, atingimos o resultado do estabelecimento do padrão de elaboração de interfaces, que segue as abordagens layout-design e navegação-visão. Além desse resultado, com este trabalho, pudemos avaliar a opção da utilização de template para a descrição de espaços virtuais do MOrFEu que foi proposta em [MENEZES, 2008].

7.1 TRABALHOS FUTUROS

A arquitetura aqui proposta é abrangente e tem potencial para cobrir os principais aspectos esperados de uma camada de apresentação. Embora a ferramenta apresentada possua limitações inerentes ao tempo de desenvolvimento, podemos vislumbrar diversos aprimoramentos e aprofundamentos. Possivelmente um dos mais importantes seja o desenvolvimento de um componente baseado em interfaces inteligentes que, baseadas em estatísticas de escolhas dos usuários, possam fornecer sugestões de templates para novos espaços virtuais.

Podemos ainda aplicar QoS (*Quality of Service*) na apresentação, o que permitiria por exemplo, templates auto-ajustáveis em relação a um mecanismo de busca. Inicialmente, no resultado da busca, seria mostrado um número determinado de resultados, que pode ser incrementado/decrementado de acordo com o tempo de navegação ou do conteúdo pesquisado.

Futuramente poderemos desenvolver ainda mecanismos para relato textual de templates. Uma ferramenta do tipo “*wizard*” que permita elaborar os templates sob uma ótica descritiva.

As possibilidades são inúmeras no que diz respeito à evolução do editor e disposição ferramental acumulativa. Contudo, existem alguns trabalhos futuros ligados à experimentação do modelo proposto aqui. Acreditamos que a realização de testes utilizando diferentes usuários participando de diferentes ambientes, seja trabalho futuro mais próximo no contexto dos templates.

REFERÊNCIAS

AHLBERG, C., SHNEIDERMAN, B. **Visual information seeking : tight coupling of dynamic query filters with starfield displays**. In: Readings in information visualization: using vision to think. San Francisco (CA): Morgan Kaufinann Publishers. xvii, 686 p. pp. 244-50. 1999.

BLACK, E. W, BECK, D., DAWSON, K., JINKS, S., DIPIETRO, M. **The other side of the LMS: Considering implementation and use in the adoption of an LMS in online and blended learning environments**. TechTrends. Springer Boston. pag. 35-39. ISSN 8756-3894 (Print) 1559-7075 (Online), 2007

BODART, F., HENNEBERT, A., LEHEUREUX, J., PROVOT, I., VANDERDONCKT J. **A Model-Based Approach to Presentation: A Continuum From Task Analysis to Prototype**. Eurographics. pp. 25-39, 1994

CAMPANA, V. F, MENEZES, C. S., TAVARES, O. L. **Veículo de Comunicação - Uma abordagem para o desenvolvimento rápido de Recursos Digitais para elaboração de Arquiteturas Pedagógicas**. IV WAPSEDI, Florianópolis – SC, 2009.

CERI, S., FRATERNALI, P., BONGIO, A., BRAMBILLA, M., COMAI, S., MATERA, M.: ***Design*ing Data-Intensive Web Applications**. Morgan Kaufmann (2003)

COELHO, M., MENEZES, C. S., PESSOA, J. M. **Timoneiro: Um ambiente de apoio a construção de Mapas Conceituais a partir de um banco de perguntas e respostas. in: XII Simpósio Brasileiro de Informática na Educação (SBIE)**. Vitória – ES, 2001.

CONFORTO, D., SANTAROSA, L.M.C. **ACCESSIBILITY: Discussing Human-Computer Interaction on the Web** In: Computers and Education: Towards a Lifelong Learning Society ed. Espanha: Kluwer Academic Publischer, 2003.

FIALA, Z., FRASINCAR, F., HINZ, M., HOUBEN, G. J., BARNA, P., MEISSNER, K. **Engineering the Presentation Layer of Adaptable Web Information Systems**. In: International Conference on Web Engineering (ICWE). Springer Verlag (July 2004), p. 459-472.

FOLEY, J., KIM, W. C., KOVACEVIC, S., MURRAY K. **UIDE - An Intelligent User Interface *Design* Environement.** In: J.Sullivan & Tyler (Eds) Architectures for intelligent interfaces: elements and prototypes. Reading MA: Addison-Wesley, pp. 339-385, 1991

FURTADO, M. E. S., **Integrando Fatores Humanos no Processo de Desenvolvimento de Interfaces Homem-Computador Adaptativas**, II Workshop sobre Fatores Humanos em SC, Campinas, SP, Brasil, 1999.

GAVA, T. B. S.; MENEZES, C. S. **Moonline: um ambiente de aprendizagem cooperativa baseado na Web para apoio às atividades extraclasse. XII Simpósio Brasileiro de Informática na Educação (SBIE)**. Vitória – ES, 2001.

GOMEZ, J., CACHERO, C. **OO-H Method: Extending UML to Model Web Interfaces**. Idea Group Publishing (2003) 144-173

GUEDES, G. **Interface Humano Computador: prática pedagógica para ambientes virtuais.** Disponível em: <http://www.ufpi.br/uapi/conteudo/disciplinas/video/livro_gildasio.pdf>. 2009. Acesso em: 25 out. 2011.

GUIA – **Grupo Português pelas iniciativas de Acessibilidade**. Disponível em: <http://www.acessibilidade.net/>. Acesso em: Jan 2012.

HEWETT, T. T., BAECKER, R., CARD, S., CAREY, T., GASEN, J., MANTEI, M., PERLMAN, G., STRONG, G., VERPLANK, W. **ACM SIGCHI Curricula for Human-Computer Interaction**. New York: The Association for Computing Machinery, 1992.

ISO 9241, Part 11 **Ergonomic requirements for office work with visual display terminals, Part 11 Usability Statements**. Draft International Standard ISO 9241-11,1993

J. NIELSEN, R. MOLICH, **Heuristic Evaluation of User Interfaces.** In: ACM CHI'90 Conf. Seattle, WA, 1-5 April, 249-256, 1990

JACOBSON I., BOOCH G., & RUMBAUGH J. **The Unified Software Development Process**. Addison Wesley. 1999

KERER, C., KIRDA, E. ***Layout*, Content and Logic Separation in Web Engineering**. In: Proceeding Web Engineering, Software Engineering and Web Application Development. London, UK, 2001

KIRDA, E., KERER, C. **MyXML: An XML Based Template Engine For The Generation of Flexible Web Content**. In: Proceedings of WebNet 2000 - World Conference on the WWW and Internet, San Antonio, Texas, USA. 2000.

KOCH, N., KRAUS, A., HENNICKER, R.: **The Authoring Process of The Uml-Based Web Engineering Approach**. In: First International Workshop on Web-Oriented Software Technology. 2001

KRZYSZTOF, G., WELD, D. S. **SUPPLE: Automatically generating user interfaces**. IN: IUI '04 Proceedings of the 9th international conference on Intelligent user interfaces ACM New York, USA. 2004.

MENEZES, C. S., CURY, D., CASTRO Jr, A. N. **An Architecture of an Environment for Cooperative Learning (AmCorA)**. Proceedings of International Conference on Engineering and Computer Education (ICECE), São Paulo – SP, 2000.

MENEZES, C. S., NEVADO, R. A., CASTRO, A. N. Jr., SANTOS, L. N. **MOrFEU – Multi-Organizador Flexível de Espaços VirtUais para Apoiar a Inovação Pedagógica em EAD** In: Simpósio Brasileiro de Informática na Educação - SBIE 2008, Fortaleza-CE, Nov/2008.

NASCIMENTO, C. V. **Diretrizes de acessibilidade para o MOrFEu.** Tese Mestrado, Universidade Federal Do Espírito Santo, Vitória, 2011.

NEVADO, R. A., MENEZES, C. S., VIEIRA R. R. M. Jr. **Debate de Teses – Uma Arquitetura Pedagógica**. In: XXII Simpósio Brasileiro de Informática na Educação (SBIE), Aracajú, AL, 2011.

ORLANDO, R. **Dispositivo da Interface: Um Estudo Sobre Tecnologias da Informação.** Tese de Doutorado, USP, São Paulo, SP, 2006.

PASTOR, O., FONS, J., PELECHANO, V. **OOWS: A Method to Develop Web Applications from Web-Oriented Conceptual Models**. In: International Workshop on Web Oriented Software Technology (IWWOST). (2003) 65-70

PERUCH, L. A., MENEZES, C. S. **Aplicando Ontologia de Colaboração na Modelagem de Ambientes Virtuais de Aprendizagem**. SBIE 2010. João Pessoa, PB, 2010

PESSOA, J. M., NETTO, H. V., MENEZES, C. S. **FAmCorA: um *FrameWork* para a construção de ambientes cooperativos inteligentes de apoio a aprendizagem na Internet baseado em web services e agentes**. XIII Simpósio brasileiro de Informática na Educação (SBIE). Leopoldo – RS, 2002.

PETERMANN, R. J., BELLIN, F., KROTH E. **Uma Ferramenta para Geração de Interfaces de Sistemas de Informação em Ambiente Web**. In: XV Simpósio Brasileiro de Engenharia de Software, Rio de Janeiro, RJ. 2001

PIAGET, J. **Les relations entre l'intelligence et l'affectivité dans le developpement de l'enfant. In Les Émotions.** Niestlé, pag. 75-95. 1989

PIRES, J. P. **O perfil dos usuários de caixas-automáticos em agências bancárias na cidade de Curitiba.** Tese de Mestrado. Universidade Federal do Paraná, 1996.

RANGEL, V. G, BELTRAME, W. A. R., CURY, D., MENEZES C. S. **MOrFEu: Towards the *Design* of an Environment for Flexible Virtual Spaces Organization**. WCCE – World Conference on Computer in Education. Bento Gonçalvez, RS – Brazil, 2009.

RANGEL, V. G. **VCom: Uma abordagem para modelagem de ambientes colaborativos.** Tese Mestrado, Universidade Federal Do Espírito Santo, Vitória, 2011.

SALES, M. B., CYBIS, W. A. **Desenvolvimento de um checklist para a avaliação de acessibilidade da web para usuários idosos**. In: CLIHC '03: Proceedings of the Latin American conference on Human-computer interaction, pp. 125 – 133, Brasil, 2003.

SANTOS, L. N., CASTRO A. N. Jr., MENEZES C. S. **MOrFEu: Criando Ambientes Virtuais Flexíveis na Web para Mediar a Colaboração**. Congresso Ibero-americano de Informática Educativa, Santiago, Chile, 2010

SANTOS, O. E. dos, **Ambientes virtuais de aprendizagem: por autorias livres, plurais e gratuita**, Revista da FAEEBA - Educação e Contemporaneidade, v. 11, n. 18, p. 425-435, Salvador, jul./dez. 2002

SCHIMIGUEL, J., MELO, A. M., BARANAUSKAS, M. C. C. et al. **Accessibility as a Quality Requirement: Geographic Information Systems on the Web**. In: CLIHC '05: Proceedings of the 2005 Latin American conference on Human-computer interaction. ACM International Conference Proceeding Series; Vol. 124, pp. 8 – 19, México, 2005.

SCHWABE, D. ROSSI, G. BARBOSA, S. D. J. **Systematic hypermedia application de-sign with OOHDM**. In: Hypertext '96, The Seventh ACM Conference on Hyper-text, Washington DC, 1996, ACM (1996) 116-128

SHNEIDERMAN, B. **Direct manipulation for comprehensible, predictable and controllable user interfaces**. In: 2"d International Conference on Intelligent User Interfaces. Orlando, Florida, USA. 1997.

SHNEIDERMAN, B. **Direct manipulation: a step beyond programming languages**. IEEE Computer l6(8) : 57-69, August. (ISSN 0018-9162). 1983

SZEKELY P., LUO, P., NECHES, P. **Beyond Interface Builders: Model-Based Interface Tools**. In CHI '93: Proceedings of the INTERACT '93 and CHI '93 conference on Human factors in computing systems, 383-390. New York, NY, USA. 1993

TURKLE, S. **A Vida no Ecrã: A Identidade na Era de Internet**. Lisboa: Relógio d'Água Editores. Tradução: Paulo Faria. 482 p. (Coleção A Sociedade Digital). 1997

UJITA., H. **Human Characteristics of Plant Operation and Man-Machine Interface**, Reliability engineering and system safety. Vol 38, pp. 119-124, 1992.

VALENTE, C. E. **Padrões de Interação e Usabilidade.** Dissertação de Mestrado Profissional, UNICAMP, Campinas, SP, 2004.

VIEIRA, R. R. M. V. Jr. **Uma Arquitetura de Software para MOrFEu: Apoiando a Realização de Arquiteturas Pedagógicas em Espaços Virtuais Colaborativos.** Tese Mestrado, Universidade Federal do Espírito Santo, Vitória, 2011.

WINCKLER, M., PIMENTA, S. **Avaliação de Usabilidade de Sites Web**. Disponível em: <http://www.funtec.org.ar/usabilidadsitiosweb.pdf>, Julho, 2000. Acesso em: 13 nov. 2011.

ZEVE, C. M. D. **Modelo cooperativo construtivista para autoria de cursos à distância usando tecnologia de Workflow.** Tese Doutorado, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2003.

APÊNDICE I. TECNOLOGIAS UTILIZADAS NA IMPLEMENTAÇÃO DO EDITOR

CSS[6]: *Cascading Style Sheets* é um mecanismo simples para adicionar estilo (por exemplo, fontes, cores, espaçamentos) aos documentos Web. Seu principal benefício é prover a separação entre o estilo e o conteúdo de um documento. Ao invés de colocar a formatação dentro do documento, é criada uma ligação (link) para o arquivo que contém os estilos. Dessa forma pode-se alterar a aparência de várias páginas, modificando apenas um arquivo.

HTML[7]: *HyperText Markup Language* é a linguagem de publicação de sites, e é um dos principais componentes da plataforma da Web aberta. A primeira versão do HTML foi descrito por Tim Berners-Lee no final de 1991. A recomendação atual para W3C HTML é HTML 4.01, publicada no final de 1999. No entanto, há um trabalho intensivo para definir sua próxima versão, HTML5.

XML[8]: *Extensible Markup Language* (XML) é um simples formato de texto muito flexível derivado do SGML (ISO 8879). Originalmente concebido para enfrentar os desafios para publicação eletrônica. XML está desempenhando um papel cada vez mais importante na troca de dados na Web e em outros lugares. A principal característica do XML é que linguagens desconhecidas e de pouco uso também podem ser definidas sem maior trabalho e sem necessidade de serem submetidas aos comitês de padronização.

XSL[9]: *Extensible Stylesheet Language* consiste em uma família de recomendações para a definição de transformação de documentos XML e apresentação. É composto de três partes: *(i)* XSL *Transformations* (XSLT): uma linguagem para transformar XML; *(ii)* XML *Path Language* (XPath): uma linguagem de expressão usada por XSLT (e muitas outras linguagens) para

[6] Site oficial: http://www.w3.org/Style/CSS/
[7] Site oficial: http://www.w3.org/html/
[8] Site oficial: http://www.w3.org/XML/ e http://www.w3.org/standards/xml/
[9] Site oficial: http://www.w3.org/Style/XSL/

acessar ou se referir a partes de um documento XML e *(iii)* Objetos de formatação XSL (XSL-FO): um vocabulário XML para especificar semânticas

PHP[10]: Trata-se de acrônimo recursivo para "*Hypertext Preprocessor*" e é uma linguagem de script amplamente utilizada de propósito geral que é especialmente adequada para o desenvolvimento Web e pode ser incorporada em HTML. É uma linguagem extremamente modularizada, o que a torna ideal para instalação e uso em servidores web. Alguns de seus módulos são introduzidos como padrão em novas versões da linguagem. É muito parecida, em tipos de dados, sintaxe e funções, com as linguagens C e C++.

JavaScript[11]: É uma linguagem de script baseada em ECMAScript padronizada pela *Ecma International* nas especificações ECMA-262 e ISO/IEC 16262. Atualmente é a principal linguagem para programação *client-side* em navegadores web para adicionar interatividade a páginas HTML. Possui suporte à programação funcional, é extremamente leve e apresenta recursos como fechamentos e funções de alta ordem comumente indisponíveis em linguagens populares como Java e C++.

MYSQL[12]: Trata-se de um sistema de gerenciamento de banco de dados (SGBD), que utiliza a linguagem SQL como interface e pode ser executado em mais de 20 plataformas, incluindo Linux, Windows, Mac, Solaris, IBM AIX. MySQL supre quase todas as funcionalidades que os SGBDs atuais pagos fornecem, como: controle transacional, *Triggers*, *Cursors* (*Non-Scrollable* e *Non-Updatable*), *Stored Procedures* e *Functions*. O sucesso do MySQL deve-se em grande medida à fácil integração com o PHP.

Apache[13]: O Apache HTTP Server, comumente referido como o Apache, é um software de servidor Web notável que desempenhou um papel chave no crescimento inicial da *World Wide Web*. Normalmente é executado em sistemas operacionais *UNIX-like*. Apache é desenvolvido e mantido por uma

[10] Site oficial: http://www.php.net/
[11] Site oficial: http://www.w3.org/standards/techs/js#w3c_all
[12] Site oficial: http://www.mysql.com/
[13] Site oficial: http://www.apache.org/

comunidade aberta de programadores sob a égide da *Apache Software Foundation.* O aplicativo está disponível para uma ampla variedade de sistemas operacionais, incluindo Unix, FreeBSD, Linux, Microsoft Windows etc.