5
Exemplo: modelo navegacional para conteúdo educacional

Para validação e teste do ambiente HyperDE foram desenvolvidas duas aplicações de exemplo, das quais estaremos detalhando neste capítulo apenas a segunda.

A primeira aplicação foi baseada no modelo navegacional apresentado em [Szundy, 2004] e seu desenvolvimento dentro do ambiente HyperDE serviu como prova de conceito. O modelo navegacional desta aplicação representa de forma simplificada o modelo do website do departamento de informática da PUC-Rio, através de seus alunos, professores, áreas de pesquisa e publicações.

PUC-Rio - Certificação Digital Nº 0210503/CA

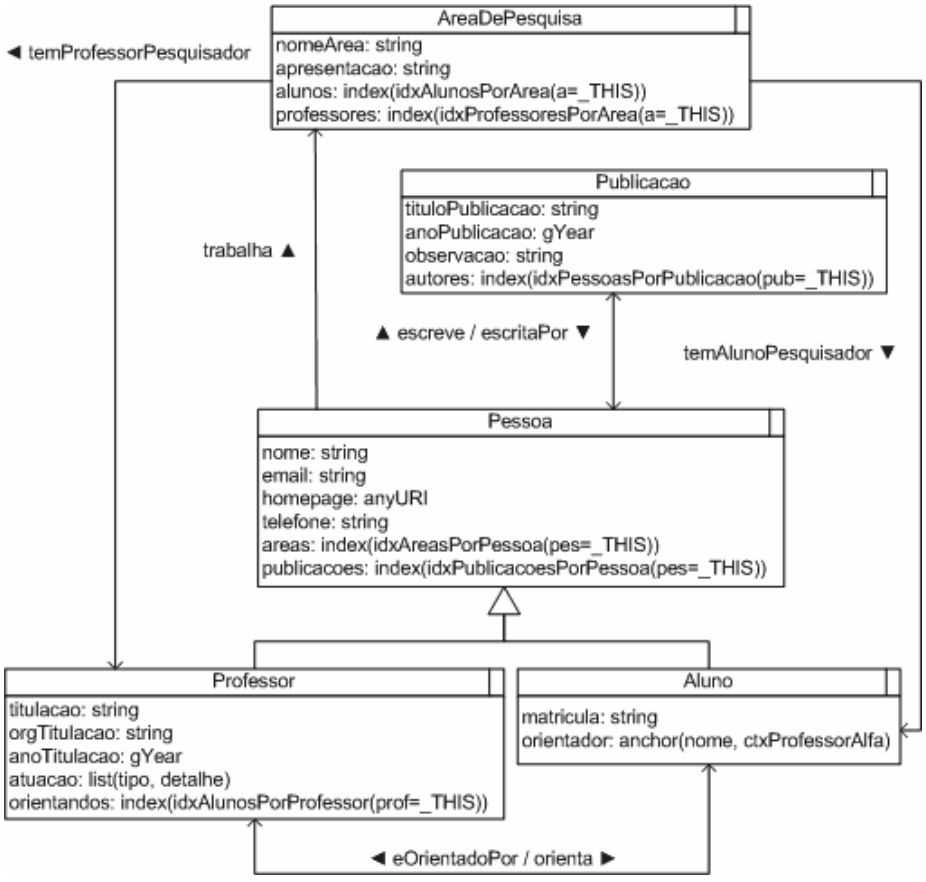

Figura 57 - Esquema navegacional para a primeira aplicação de exemplo

Como podemos ver pelo diagrama, este é um modelo bastante simples e compacto, possuindo apenas cinco classes navegacionais e sete tipos de elos. Podemos visualizar seus contextos, landmarks e estruturas de acesso a seguir:

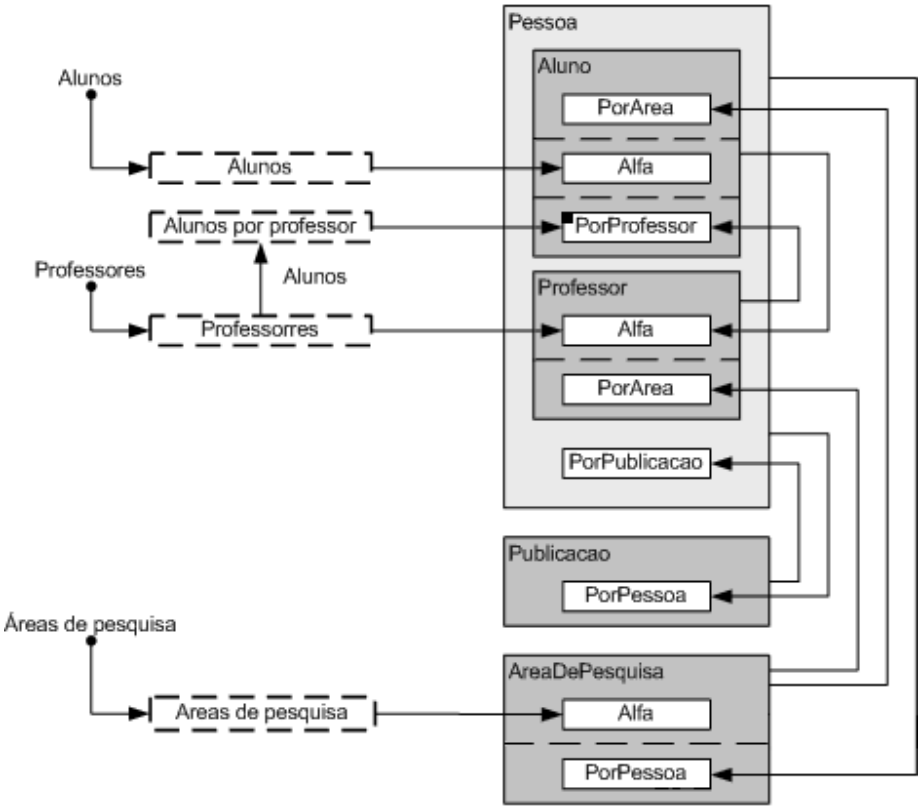

Figura 58 - Esquema de contextos navegacionais da primeira aplicação exemplo

Neste esquema de contextos podemos identificar 3 landmarks, 4 índices e 9 contextos navegacionais. Apesar de constituir um modelo de classes relativamente pequeno, o diagrama de contextos possui a diversidade dos tipos de classes, atributos, elos, contextos e índices equivalentes às aplicações tipicas encontradas em ambientes industriais, o que nos permitiu avaliar com abrangência o suporte dado pelo ambiente HyperDE a essas primitivas.

Após a construção do modelo navegacional desta aplicação de exemplo, realizamos o carregamento em lote de aproximadamente 1000 nós, em sua maioria nós derivados das classes “Aluno”, “Professor” e “Publicacao”, realizando a navegação sobre estes nós sob os contextos definidos no modelo, verificando o tempo de resposta do ambiente e sua adequação a refinamentos e modificações do modelo enquanto a aplicação era executada.

PUC-Rio - Certificação Digital Nº 0210503/CA

Para o segundo exemplo, utilizamos um modelo navegacional significativamente mais complexo e que veremos em detalhes nas seções a seguir.

5.1. Esquema de Classes Navegacionais

Para a segunda aplicação de exemplo do ambiente HyperDE, decidimos desenvolver um modelo que representasse de forma bastante liberal, uma universidade virtual, onde o foco funcional está concentrado na gestão de conteúdos educacionais criados e visualizados no próprio ambiente por usuários, classificados como alunos e professores de tal universidade.

O estilo de interação com os conteúdos da aplicação assemelha-se a de um *Wiki* [Leuf & Cunningham, 2001], onde modificações e manipulações são realizadas *"in-place"*, utilizando os controladores de retaguarda para criação e modificações dos nós.

Os principais requisitos funcionais atendidos pela aplicação estão listados a seguir:

- Estruturação Organizacional: o modelo fornece classes e elos para representação de departamentos de uma universidade virtual, assim como professores e alunos ligados a tais departamentos.
- Estruturação de Conteúdos: o modelo fornece classes e elos que permitem a estruturação flexível de conteúdos educacionais, através de hierarquização e composição, além de tipos diversos de classificação de conteúdos.
- Comentários a Conteúdos: o modelo fornece classes e elos que permitem aos visitantes adicionarem comentários aos conteúdos educacionais criados na aplicação, podendo-se utilizar deste recurso para avaliar a qualidade dos mesmos.
- Referências Web Externas de Conteúdos: o modelo fornece classes e elos que permitem a catalogação de referências web externas aos conteúdos criados na aplicação, ou seja, a indexação de páginas e websites externos aos conteúdos locais.
- Exercícios: o modelo fornece classes e elos para avaliações sobre os conteúdos educacionais apresentados pela aplicação através da

PUC-Rio - Certificação Digital Nº 0210503/CA

realização de questões de exercícios, que podem ser respondidos por alunos e corrigidos por professores.

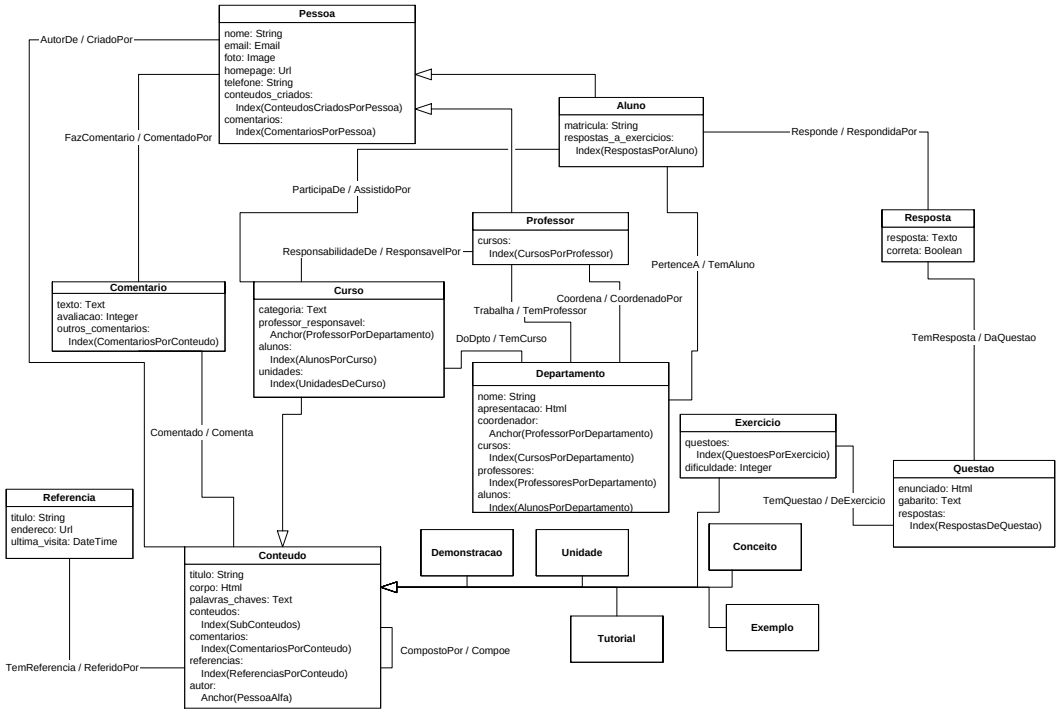

Figura 59 - Esquema de classes navegacionais para o modelo educacional da aplicação de exemplo

No diagrama acima temos uma visão geral sobre a representação do modelo de 16 classes navegacionais da aplicação, assim como os 28 tipos de elos que definem as relações entre elas.

Por este modelo podemos obter uma interpretação básica da aplicação, que listaremos a seguir (note-se que as classes navegacionais do modelo estão citadas em letra maiúscula, podendo aparecer em sua forma singular ou plural):

- Alunos e Professores são tipos de Pessoa, que estão afiliados a Departamento, sendo que Professor pode possuir um tipo de relação especial com Departamento, quando o primeiro exerce uma função de coordenação sobre o último. Ambos podem realizar autoria de conteúdos e podem também comentar conteúdos existentes. Apenas Aluno pode responder a Questões de Exercícios. Professores podem ser responsáveis por Cursos e Alunos podem assistir Cursos.

PUC-Rio - Certificação Digital Nº 0210503/CA

- Curso é um tipo de Conteúdo, assim como Unidade, Tutorial, Conceito, Exemplo, Exercício e Demonstração. Conteúdos podem ser compostos de outros Conteúdos e podem possuir Referências endereços na Web. Cursos podem pertencer a um Departamento.
- Exercício, que é um tipo de Conteúdo, possui Questões, que podem possuir Respostas de Alunos.

Praticamente todas as classes navegacionais apresentadas no diagrama anterior, possuem atributos simples, como “nome” (classe Pessoa), “foto” (classe Pessoa), “endereco” (classe Referencia), “enunciado” (classe Questao), respectivamente dos tipos de dados “String”, “Image”, “Url” e “Html”.

Da mesma forma, as mesmas classes possuem também atributos navegacionais de tipo âncora (para contexto ou índice) e de tipo índice, isto é, atributos que determinam caminhos possíveis de navegação entre os nós em diferentes contextos. Estes atributos de tipo âncora ou índice, em sua maioria, estão baseados nas relações entre os nós, formadas através dos elos definidos no modelo navegacional. Alguns exemplos:

- A classe Curso possui um atributo de tipo âncora para contexto, baseado no elo “ResponsavelPor”, cujo contexto alvo é “ProfessorPorDepartamento”, possibilitando a navegação entre um nó de tipo Curso para um nó de tipo Professor neste contexto.
- A classe Conteudo possui um atributo de tipo índice, baseado no elo “CompostoPor”, cujo índice alvo é “SubConteudos”, possibilitando a listagem e navegação a nós de tipo Conteudo que compõem determinado nó.
- A classe Conteudo possui também um atributo de tipo índice, baseado no elo “Comentado”, apontando para o índice “ComentariosPorConteudo”, possibilitando a listagem e navegação de nós do tipo Comentario, que comentam determinado Conteudo.
- A classe Departamento possui atributos de tipo índice e âncora para índice, para listagem e navegação de nós de tipo Aluno, Professor e

PUC-Rio - Certificação Digital Nº 0210503/CA

Curso, relacionados a determinado Departamento através dos elos “TemAluno”, “TemProfessor” e “TemCurso” respectivamente.

Além de atributos simples e navegacionais, algumas classes da aplicação possuem também atributos computados, como por exemplo, o atributo “correta” da classe Resposta, que compara o valor da resposta ao gabarito da questão e determina um valor booleano (verdadeiro ou falso) para o atributo do nó ou o atributo “avaliacao_media” da classe Conteudo, que calcula a média de avaliações dada ao nó através de comentários.

Por último, temos ainda classes navegacionais que implementam algumas operações, que serão vistas em maior detalhe em subseção posterior deste capítulo.

Portanto, os atributos, índices e âncoras que aparecem no esquema de classes navegacionais tornam possível deduzir algumas transições e estruturas de navegação que são oferecidas pela aplicação. No entanto, para termos uma visão mais específica sobre este aspecto veremos a seguir o esquema de contextos navegacionais, estruturas de acesso e landmarks da aplicação.

PUC-Rio - Certificação Digital Nº 0210503/CA

5.2. Esquema de Contextos Navegacionais

O diagrama a seguir apresenta todos os 22 contextos navegacionais modelados na aplicação, assim como as 12 estruturas de acesso independentes, ou seja, 12 índices que não estejam definidos como atributos de uma classe navegacional (estes podem ser visualizados no esquema de classes navegacionais apresentado anteriormente). O diagrama apresenta ainda os 4 landmarks disponíveis na aplicação.

Dentre os contextos navegacionais, temos 3 contextos formados a partir de nós de uma mesma classe navegacional, ordenados alfabeticamente, que são “DepartamentoAlfa”, “CursoAlfa” e “ConteudoAlfa”.

A grande maioria dos contextos, porém, é definida a partir de nós de uma mesma classe navegacional parametrizados pela relação entre estes nós e um nó de outra classe navegacional e que são ligados através de um tipo de elo. Exemplos deste tipo de contexto são:

- CursoPorProfessor: formado por nós da classe navegacional Curso, derivados do elo “ResponsavelPor” que é ligado a um nó da classe Professor.
- ProfessorPorDepartamento: formado por nós da classe Professor, derivados do elo “Trabalha” que é ligado a um nó da classe Departamento.

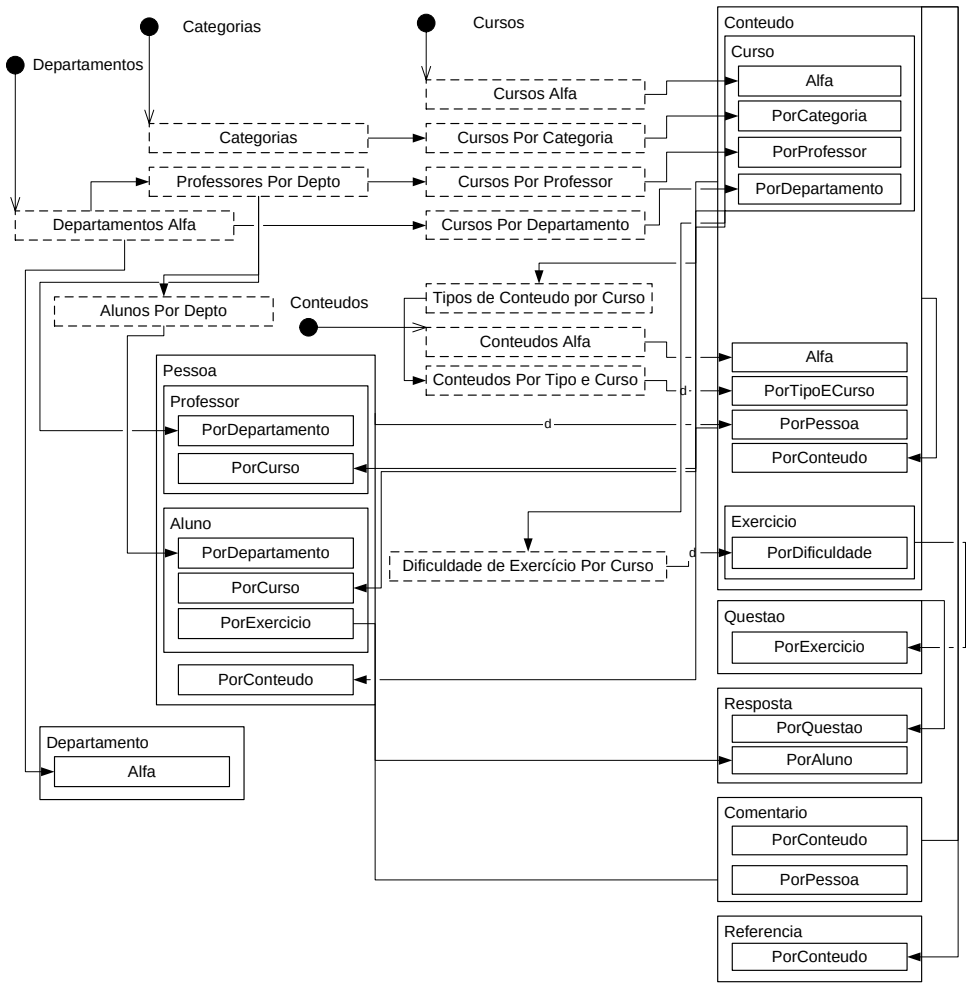

Figura 60 - Esquema de contextos, índices e landmarks para o modelo educacional da aplicação de exemplo

Existem ainda alguns contextos onde os nós que os compõem são de determinada classe navegacional e possuem em comum um determinado valor de atributo que permite selecioná-los, onde tal valor é passado como parâmetro à expressão de consulta do contexto. Um exemplo deste tipo de contexto é

PUC-Rio - Certificação Digital Nº 0210503/CA

“ConteudoPorCategoria”, onde os nós são da classe Conteudo e possuem um determinado valor do atributo “categoria”.

Um outro tipo de contexto, parecido com o descrito anteriormente, é formado por nós de determinada classe, mas que combinam ainda a característica de serem derivados a partir da relação de um tipo de elo e de determinado valor de um meta-atributo, referente ao metamodelo da aplicação, que é passado como parâmetro. O exemplo deste contexto é “ConteudoPorTipoECurso”, onde os nós que o formam são da classe Conteudo, derivados do elo “Compoe” e cuja característica comum como fator de seleção é pertencerem a determinada subclasse de Conteudo, onde a subclasse é passada como parâmetro para expressão de consulta do contexto.

Este último tipo de contexto navegacional nos revela uma qualidade interessante do ambiente HyperDE, oriunda do metamodelo SHDM e da base de dados, de fornecer a possibilidade de realizar consultas utilizando não apenas critérios de seleção baseados nas instâncias do modelo da aplicação, i.e. seus nós e valores de atributos, mas também utilizar características do metamodelo que define a aplicação, ou seja, suas classes navegacionais e as declarações de atributos, elos e operações.

Neste sentido, poderíamos imaginar a construção de uma expressão de consulta para contexto ou índice onde seria feita a seleção de nós que são derivados de subclasses de determinada classe e que possuem a declaração de determinado atributo, elo ou operação. Note que neste caso, estaríamos utilizando características do próprio esquema navegacional da aplicação para determinar o resultado da consulta e não valores de elos e atributos.

O uso de uma base de dados semântica, apoiada em RDF e RDF(s) como a utilizada no ambiente, nos permite ainda utilizar regras de inferência sobre o metamodelo e modelo da aplicação. Isto torna possível a construção de consultas ainda mais “inteligentes”, onde se faria uso de poder dedutivo da máquina de consulta para seleção dos nós sem a necessidade de declaração explícita de todos os critérios de seleção.

PUC-Rio - Certificação Digital Nº 0210503/CA

5.3. Operações

A seguir temos um diagrama, similar ao esquema de classes navegacionais já visto anteriormente, mas que ao invés de exibirmos os atributos das classes navegacionais, estamos apresentando as operações associadas a cada uma delas.

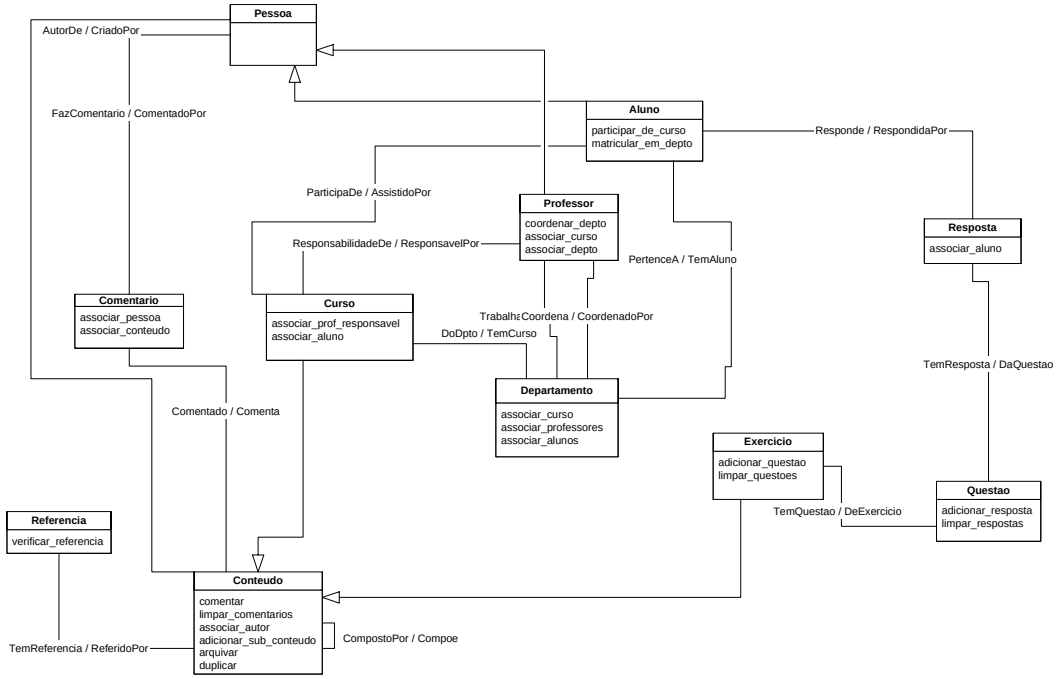

Figura 61 - Esquema de classes navegacionais exibindo as operações de cada classe

As operações implementadas na aplicação deste exemplo podem ser classificadas em dois tipos: operações que apenas auxiliam a transição navegacional para determinado controlador de retaguarda para então realizarmos uma manipulação de elos entre os nós e operações que efetivamente modificam atributos e manipulam elos do nó corrente diretamente.

O primeiro tipo de operação está presente na aplicação apenas para facilitar a operação de associação entre nós através de elos específicos na interface web. Sem este tipo de operação, esta associação aconteceria da seguinte forma:

1. Navegamos para um nó, ao qual desejamos associar outro nó através de determinado elo.

PUC-Rio - Certificação Digital Nº 0210503/CA

2. Alternamos para o metamodelo da aplicação e ativamos o controlador de retaguarda “NodeController” através da âncora da barra superior da interface web para editar o nó corrente.
3. Localizamos dentre os tipos de elos disponíveis para aquele nó o elo pelo qual desejamos realizar a associação.
4. Ativamos a funcionalidade para adicionar novo valor de elo para o tipo de elo selecionado.
5. Efetivamos a associação do nó alvo ao nó de origem através do elo escolhido.
6. Retornamos ao controlador de navegação da aplicação e visualizamos os possíveis efeitos da associação recém-realizada.

Utilizando a operação da classe destinada especificamente a fazer tal associação, precisamos realizar apenas os últimos 3 passos para efetuar a associação. Dessa forma, a operação apenas nos posiciona diretamente na ação de adicionar novo valor de elo de tipo já pré-determinado pela operação.

Alguns exemplos deste tipo de operação são:

- “associar_autor” (classe Conteudo): visando a criação de elo do tipo “CriadoPor” entre um nó da classe Conteudo e um nó da classe Pessoa.
- “associar_curso” (classe Departamento): visando a criação de elo do tipo “TemCurso” entre um nó da classe Departamento e um nó da classe Curso.
- “associar_prof_responsavel” (classe Curso): visando a criação de elo do tipo “ResponsabilidadeDe” entre um nó da classe Curso e um nó da classe Professor.

A existência deste tipo de operação se faz necessária apenas por termos decidido utilizar os próprios controladores de retaguarda do ambiente para manipular os nós e elos da aplicação.

Enquanto que esta decisão nos traz o benefício de manter o exemplo simples e implementável em curto espaço de tempo, seria inviável termos tal dependência entre o ambiente de desenvolvimento e a aplicação em si para um ambiente de

PUC-Rio - Certificação Digital Nº 0210503/CA

produção. Desta forma, seria necessária a implementação de operações completas para realizar toda a manipulação de nós e elos e visões customizadas que apresentassem os elementos de interface adequados para tais ações. No entanto, os atuais ambientes *"Wiki"* possuem este tipo de facilidade, pois são voltados para a edição dinâmica da aplicação, por um grupo de pessoas.

Um segundo tipo de operação implementada na aplicação efetivamente pode modificar atributos ou criar novos nós associados ao nó de origem, como também podem opcionalmente realizar uma transição navegacional para os controladores de retaguarda. Alguns exemplos deste tipo de operação são:

- "verificar_referencia" (classe Referencia): realiza uma verificação da URL da referência em questão, contida no atributo "endereco" do nó, para determinar se esta URL ainda é válida/existente. Em caso positivo, atualiza o atributo "ultima_verificacao_em" com a data e hora da verificação.
- "comentar" (classe Conteudo): cria um novo nó da classe Comentario, associa este nó ao nó corrente (da classe Conteudo) através do elo "Comenta" e realiza a transição navegacional para a tela de edição do nó da classe Comentario, recém criado e associado.
- "duplicar" (classe Conteudo): cria um novo nó da classe Conteudo, duplicando os valores de atributos e elos do nó original e realizar a transição navegacional para tela de edição do nó duplicado.
- "limpar_respostas" (classe Questao): exclui todos os nós do tipo "Resposta" associado ao nó corrente (da classe Questao), através do nó "TemResposta".
- "arquivar" (classe Conteudo): atualiza o atributo "arquivado_em" do nó corrente com o valor da data e hora atuais.

Vale frisarmos que todas as transições navegacionais efetuadas pelas operações implementadas nas classes navegacionais da aplicação estão definidas, na realidade, na declaração da âncora de ativação da operação, que é realizada nos templates de visões customizadas, através da função "node_op" já vista anteriormente.

PUC-Rio - Certificação Digital Nº 0210503/CA

O código Ruby definido nas operações não tem poder de executar transições navegacionais e tais transições foram descritas nesta seção como se implementadas pelas operações apenas para facilitar o entendimento do efeito total de cada operação.

5.4. Visões

A definição de algumas visões customizadas dentro da aplicação de exemplo visa obter uma visualização mais elaborada para os nós em contexto e índices da aplicação através de templates especificamente definidos para a exibição de nós das classes navegacionais e índices da aplicação.

Outro benefício adicional do uso de visões customizadas é a possibilidade de dar suporte a classes em contexto. Classes em contexto, conforme visto anteriormente, é um padrão de projeto baseado em decorador, que define a modificação de atributos e operações de classes navegacionais, dependendo do contexto navegacional em que suas instâncias, os nós, são navegados.

Originalmente, os métodos OOHDM e SHDM definem classes em contexto como um aspecto pertencente à etapa de modelagem navegacional, e portanto, se suportado fielmente, o ambiente HyperDE deveria fazê-lo na camada de modelo navegacional. No entanto, analisamos que se suportado dessa forma pelo ambiente, seria introduzido um grau de complexidade adicional à camada de modelo navegacional do framework do ambiente, com poucos benefícios práticos.

Dessa forma, decidimos que o ambiente deveria dar suporte a classes em contexto apenas na camada de visão, através de templates específicos em visões customizadas associadas a classe e contexto. Nestas visões podemos determinar quais atributos e operações de determinada classe navegacional em determinado contexto estarão visíveis e disponíveis através da interface com o usuário da aplicação.

Foram definidas as seguintes visões customizadas para a aplicação de exemplo:

- Visões Genéricas

PUC-Rio - Certificação Digital Nº 0210503/CA

  - Layout: definido o layout global da aplicação, onde é carregada a folha de estilo CSS da aplicação e são definidos cabeçalho e rodapé.
  - BarraDeNavegacao: template para desenho do componente de barra de navegação em contexto.
  - Breadcrumb: template para desenho do componente de rastro de navegação.
  - IndiceDeContexto: template para desenho do componente de índice de contexto.
  - TituloDeContexto: template para desenho do componente de título da página em navegação de nós em contexto.
- Visões de Índices
  - Departamentos: especificamente definido para a exibição do índice “Departamentos Alfa” por tratar-se da página de entrada da aplicação.
  - Cursos: definido para todos os índices cujos elementos são nós da classe Curso.
  - Pessoas: definido para todos os índices cujos elementos são nós das classes Professor e Aluno.
  - Indices: definido para todos os outros índices da aplicação.
- Visões de Classe e Contexto
  - Curso: definido para navegação em nós da classe Curso, em qualquer contexto navegacional.
  - Conteudo: definido para navegação em nós da classe Conteudo, em qualquer contexto navegacional.
  - Professor: definido para navegação em nós da classe Professor, em qualquer contexto navegacional.
  - Aluno: definido para navegação em nós da classe Aluno, em qualquer contexto navegacional.
  - Departamento: definido para navegação em nós da classe Departamento, em qualquer contexto navegacional.
  - Exercicio: definido para navegação em nós da classe Exercicio, em qualquer contexto navegacional.

PUC-Rio - Certificação Digital Nº 0210503/CA

- Questao: definido para navegação em nós da classe Questao, em qualquer contexto navegacional.
- Resposta: definido para navegação em nós da classe Resposta, em qualquer contexto navegacional.
- Comentario: definido para navegação em nós da classe Comentario, em qualquer contexto navegacional.
- Referencia: definido para navegação em nós da classe Referencia, em qualquer contexto navegacional.
- ExercicioPorDificuldade: definido para navegação em nós da classe Exercicio, no contexto navegacional "ExercicioPorDificuldade".
- CursoPorDepartamento: definido para navegação em nós da classe Curso, no contexto navegacional "CursoPorDepartamento".
- ComentarioPorPessoa: definido para navegação em nós da classe Comentario, no contexto navegacional "ComentarioPorPessoa".

Como podemos ver pela lista acima, foram definidas visões customizadas de todos os tipos disponíveis no ambiente:

- genéricas, para definição de layout e componentes de interface reutilizáveis
- de índices, para exibição de índices utilizando templates específicos para determinados índices, como "Departamentos Alfa", e genéricos para os índices restantes da aplicação.
- de classe navegacional e contexto, onde foram criados templates específicos para exibição de nós em cada classe navegacional disponível na aplicação, de forma mais genérica, independente do contexto navegacional, assim como templates específicos tanto para a classe navegacional como para o contexto em que instâncias dessa classe serão exibidos, implementando assim o aspecto de classes em contexto.

PUC-Rio - Certificação Digital Nº 0210503/CA

Para elaboração destes templates, foi feito uso extensivo das funções de auxilio da camada de visão, já descritas anteriormente.

5.5. Implementação

Inicialmente, realizamos um levantamento de requisitos simplificado, tentando manter a aplicação em um nível de complexidade baixo, mas que no entanto, fosse capaz de demonstrar todas as capacidades do ambiente HyperDE.

Após definirmos os requisitos e realizarmos a modelagem conceitual, passamos à fase de modelagem navegacional com apoio direto do ambiente. A etapa de modelagem conceitual foi feita manualmente, sem apoio do HyperDE, mas por tratar-se de uma aplicação razoavelmente simples, houve pouca diferença entre os modelos conceitual e navegacional, e dessa forma, pouca necessidade de mapeamento entre os dois modelos. O ambiente HyperDE atualmente não dá qualquer suporte à modelagem conceitual e ao mapeamento entre o modelo conceitual e o navegacional, capacidades que poderão ser adicionadas futuramente.

O modelo navegacional foi inicialmente diagramado em ferramenta gráfica visual (Microsoft Visio) e depois realizado o mapeamaento manual do modelo gráfico para uma representação textual, utilizando o formato suportado pela ferramenta de importação de modelos navegacionais fornecida pelo ambiente. A partir daí já pudemos executar a aplicação na interface Web e colher as informações para executar os refinamentos necessários de forma interativa.

Os detalhes restantes do modelo navegacional foram implementados utilizando a interface Web do ambiente. Utilizamos a interface Web em conjunto com a interface de console para refinar o modelo até seu estágio final.

Pela interface Web pudemos alimentar o ambiente com nós, atributos e elos, com o objetivo de rapidamente podermos testar a execução da aplicação utilizando os templates nativos do ambiente.

Após consolidarmos o modelo navegacional e termos dados suficientes para validá-lo, passamos a etapa de elaboração das visões customizadas da aplicação, onde foram definidos diretamente os templates específicos para cada classe e índices da aplicação. Portanto, não houve, nesta etapa, a necessidade de definição

PUC-Rio - Certificação Digital Nº 0210503/CA

abstrata das visões por estarmos produzindo templates, utilizando as funções auxiliares do ambiente, que ficaram muito próximos de um nível abstrato.

Realizando a construção das visões customizadas no próprio ambiente, tivemos a capacidade de refinar e ajustar os templates rapidamente, pois os efeitos destes ajustes podiam ser vistos instantaneamente.

Após o refinamento das visões da aplicação até seus estágios finais, havíamos completado também a etapa de implementação da aplicação e dessa forma, concluímos o desenvolvimento da aplicação de exemplo.

Todas as etapas do desenvolvimento apoiadas pelo ambiente HyperDE foram completadas de forma altamente interativa, podendo, dessa forma, obter-se uma visibilidade constante sobre os progressos obtidos a cada passo do processo.

PUC-Rio - Certificação Digital Nº 0210503/CA