4
Modelagem de Interface

A modelagem de uma interface visa representar todas as trocas de informação que podem ocorrer entre o usuário e a aplicação. Essas trocas de informação são especificadas no decorrer da modelagem da aplicação. Ao se utilizar a metodologia SHDM para modelar uma aplicação hipermídia, desenvolve-se o modelo conceitual e, posteriormente, faz-se um mapeamento para o modelo navegacional da aplicação. Pode-se identificar quais são os objetos navegacionais e como eles estão organizados por meio do modelo navegacional, porém, não se sabe como serão apresentados. Desta forma, torna-se necessário desenvolver um modelo de interface para representar como serão apresentados esses objetos. O modelo de interface especifica os objetos perceptíveis que estarão disponíveis ao usuário.

Os objetos de interface apresentam informações e recebem a denominação de *widgets*. Esses *widgets* permitem a interação do usuário com o sistema, através, por exemplo, da realização de ações como clicar utilizando o mouse ou digitar valores por meio do teclado.

A modelagem proposta neste trabalho se dispõe a utilizar duas ontologias: uma ontologia de widgets concretos, que representa elementos concretos de interfaces e uma ontologia de widgets abstratos, que descreve possíveis elementos abstratos e que representam os elementos concretos abstratamente. Detalhamos cada uma destas ontologias a seguir.

4.1.
Modelagem de Interface Abstrata

A modelagem de interface abstrata descreve os possíveis elementos que representam as interações entre o usuário e a aplicação. A escolha desses elementos (*widgets*) é realizada pelo designer, através do modelo navegacional e da análise de requisitos, visto que a partir deles é possível se extrair todas as necessidades de trocas de informação que precisam acontecer entre o usuário e a

aplicação. Dessa forma, o designer consegue decidir quais serão os *widgets* mais adequados para a realização dessas trocas de informação.

A abordagem de modelagem de interfaces procura separar os aspectos essenciais, independentes de tecnologia e de padrões de implementação, dos aspectos específicos de cada ambiente de execução. Os aspectos essenciais, que dizem respeito à troca de informações entre o usuário e a aplicação, são representados através do uso do vocabulário definido na Ontologia de Widgets Abstratos. Desta forma, uma interface poderá ser caracterizada abstratamente instanciando-se a Ontologia de *Widgets* Abstratos para o problema em questão.

Dada uma interface abstrata descrita na Ontologia de *Widgets* Abstratos, é possível mapeá-la numa interface concreta, composta por *widgets* concretos, que por sua vez são instâncias da Ontologia de *Widgets* Concretos. Os *widgets* concretos são diretamente traduzíveis em elementos de interface disponíveis nos ambientes de implementação comumente encontrados, como por exemplo, HTML.

4.1.1. Ontologia de *Widgets* Abstratos

A ontologia de *Widgets* abstratos tem como objetivo descrever como os objetos navegacionais serão apresentados, especificando os elementos perceptíveis que estarão disponíveis para o usuário.

A ontologia foi desenvolvida em linguagem OWL (*Ontology Web Language*), com o auxilio da ferramenta Protegé [16]. A ferramenta utilizada para realizar a validação dessa ontologia foi a OWL *Validator* [17]. A ontologia é composta por 11 conceitos, que por sua vez representam os elementos da ontologia de *widgets* concretos. Esses conceitos possuem uma semelhança com as primitivas do diagrama de interação com o usuário (*User Interaction Diagram*-UID) [24], pois dão suporte às entradas do usuário (seja um item de dado ou uma estrutura fornecida pelo próprio usuário), às saída do sistema (item de dado ou estrutura retornada pelo sistema) e às operações que são solicitadas pelo usuário. Acredita-se que, através dos conceitos definidos nessa ontologia, é possível conseguir a representação de todas as interações do usuário com o sistema.

A interface abstrata é uma instância dessa ontologia, representando uma composição dos seus conceitos. Esses conceitos podem representar um ou mais

elementos de interface concreta, ou seja, *widgets* concretos. Na figura 13, é descrita a ontologia de *widgets* abstratos.

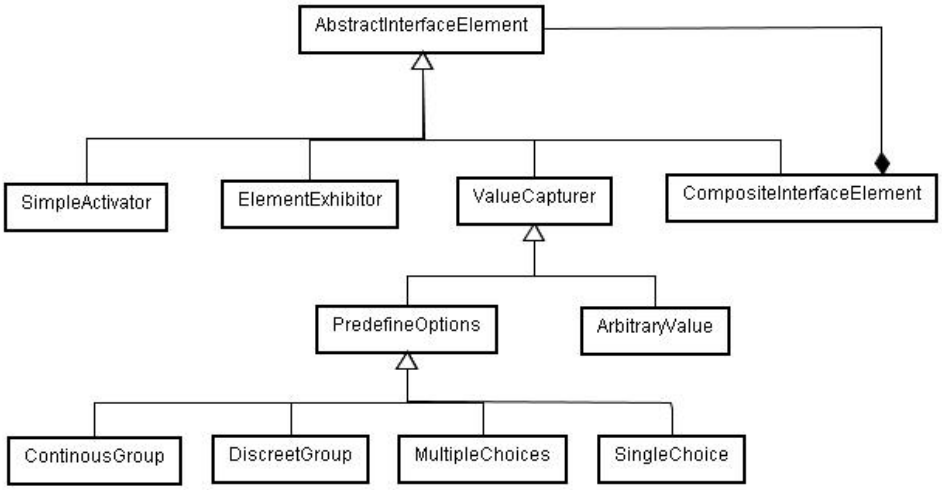

figura 13 - Ontologia de *Widgets* Abstratos

4.1.1.1. Classes da Ontologia

As classes dessa ontologia representam um ou mais elementos abstratos das interfaces das aplicações hipermídia:

- *AbstractInterfaceElement*: esta classe é composta por 4 subclasses que definem os possíveis elementos abstratos. Estes elementos representam os possíveis tipos de interações entre o usuário e o sistema.
    - *SimpleActivator*: esta classe representa qualquer elemento capaz de reagir a eventos externos – caso típico do clicar o mouse – que possua um evento associado a ele, tais como ativar um link, um botão de ação, o envio de um e-mail, dentre outros
    - *ElementExhibitor*: esta classe representa elementos que exibem algum tipo de conteúdo, como, por exemplo, um texto ou uma imagem;
    - *ValueCapturer*: esta classe representa os elementos capazes de capturar um valor, como, exemplo, as "Caixas de textos" e os elementos do tipo selecionador como: *Check Box*, *Radio Button*, entre outros. Esta classe generaliza duas classes distintas:
        - *ArbitraryValue*: permite que o usuário insira dados através do uso do teclado. Pode representar um

campo de formulário, ou seja, uma caixa de texto. O valor a ser capturado por esse elemento é desconhecido a priori.

- *PredefineOptions*: permite a seleção de um subconjunto a partir de um conjunto de valores pré-definidos; muitas vezes, esse subconjunto poderá ser unitário. As especializações dessa classe são:
  - *ContinousGroup*: permite a seleção de um valor de um conjunto infinito de valores;
  - *DiscreetGroup*: permite a seleção de um valor de um conjunto de valores enumeráveis.;
  - *MultipleChoices*: permite a escolha de mais de um elemento de um conjunto enumerável;
  - *SingleChoice*: permite a escolha de apenas um elemento de um conjunto enumerável.

- *CompositeInterfaceElement*: representa uma composição dos elementos abstratos citados acima.

- *AbstractInterface:* define a interface abstrata final que é representado por uma composição dos elementos da classe "AbstractInterfaceElement".

4.1.1.2. Propriedades da Ontologia

A ontologia possui 11 propriedades, divididas em *ObjectProperty* e *DatatypeProperty*.

- *ObjectProperty:*
  - *hasInterfaceElement*: indica os elementos da classe "AbstractInterfaceElement" que compõem os elementos do tipo "CompositeInterfaceElement" e "AbstratctInterface".
    Domínio: AbstractInterface e CompositeInterfaceElement.
    Contra-domínio: AbstractInterfaceElement.
  - *targetInterface*: indica qual será a instância da classe "AbstractInterface" a ser criada quando esse elemento for ativado. Essa instância representa uma interface abstrata.
    Domínio: SimpleActivator e CompositeInterfaceElement.

Contra-domínio: AbstractInterface.

- *mapsTo*: indica o elemento, da ontologia de *widgets* concretos, que será mapeado no elemento da ontologia de *widgets* abstrato. Esta propriedade está presente em todas as classes da ontologia de *Widgets* Abstratos.

  Domínio: Classe "ConcreteInterfaceElement" da Ontologia de *Widgets* Concretos.

  Contra-Domínio: AbstractInterfaceElement.

- *DatatypeProperty*
  - *blockElement*: indica as *tags* HTML e classes CSS que serão usadas para a tradução de um elemento específico. Esta propriedade é opcional para todas as subclasses da classe "AbstractInterfaceElement".
  - *isRepeated*: é uma propriedade apenas do conceito "CompositeInterfaceElement", que indica se os elementos que compõem esse conceito irão ou não se repetir um numero arbitrário de vezes.
  - *compositionTag*: indica uma *tag* HTML. Esta propriedade pertence à classe "CompositeInterfaceElement". Ela é utilizada somente quando a propriedade "isRepeated", dessa classe, possui como valor *true*. Sua função é indicar qual a *tag* HTML que irá separar os elementos, dessa composição, que se repetem.
  - *fromAnchor*: indica qual é a âncora descrita na ontologia navegacional, que a instância de um elemento abstrato corresponde.
  - *fromContext*: indica qual é o contexto, descrita na ontologia navegacional, que a instância de um elemento abstrato pertence;
  - *fromIndex*: indica qual o índice, descrito na ontologia navegacional, que a instância de um elemento abstrato se refere;
  - *fromAttribute*: indica qual é o atributo da ontologia navegacional correspondente à instância de um elemento abstrato;
  - *fromElement*: indica qual é o elemento da ontologia navegacional correspondente à instância de um elemento abstrato;

- *visualizationText*: representa um valor que será apresentado pelo elemento concreto. Esta propriedade é utilizada apenas nos conceitos "ElementExhibitor" e "IndefiniteVariable";

4.1.2. Modelagem

Para ilustrar o uso da ontologia anteriormente descrita, será apresentado um exemplo da modelagem de uma aplicação hipermídia que utiliza a metodologia SHDM. Por não ser objeto deste trabalho, não apresentamos os passos utilizados para modelar essa aplicação, mas mostramos apenas o diagrama de contextos desta aplicação . Suponha-se que essa aplicação seja sobre um grupo de pesquisa, do qual se têm informações sobre professores, alunos e artigos. Um diagrama de contexto possível para essa aplicação é o ilustrado na figura 14.

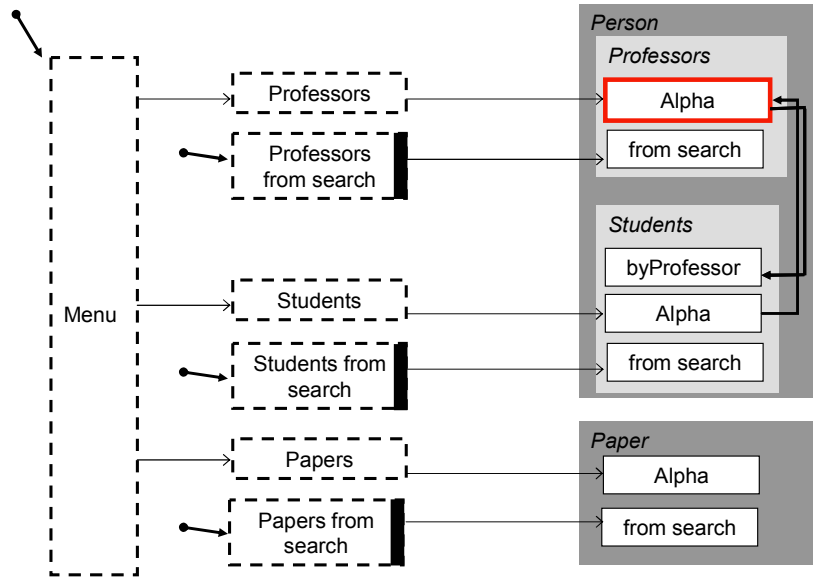

figura 14 – Diagrama de Contexto de uma Aplicação Hipermídia.

Dado o diagrama de contexto, pode-se modelar uma interface abstrata correspondente aos elementos do contexto "Professor Alpha" (contexto que permite o acesso a professores ordenados alfabeticamente pelo nome do professor). Essa interface irá apresentar dados sobre um professor específico e poderá conter também, por exemplo, um menu principal e um elemento de busca interna. Desta maneira, sabe-se que esta interface abstrata pode ter até o momento 3 áreas principais, descritas na figura 15.

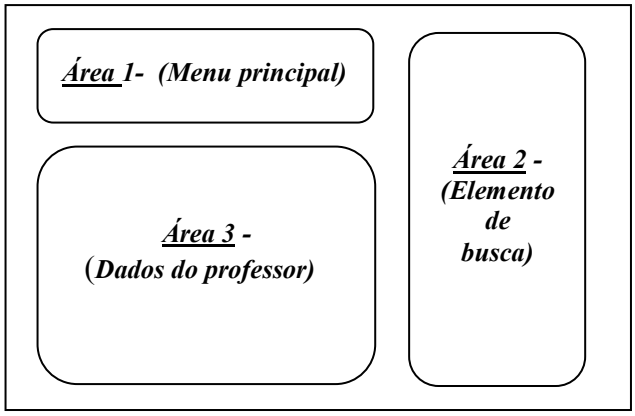

figura 15 – Representação abstrata dos possíveis elementos de interface para um objeto do contexto “Professores Alpha”.

Na área 1 da figura 15, tem-se o elemento “Menu principal”. A partir do diagrama de contextos sabe-se que este elemento corresponde a um índice. Como as entradas de um índice reagem a eventos externos, então se pode afirmar que possivelmente esse elemento será uma composição de “SimpleAcivator”s. Neste diagrama de contexto, o elemento “Menu principal” permite a navegação para professores (índice “Professors”), alunos (índice “Students”) e artigos (índice “Papers”). Desta forma, sabe-se que o elemento “Menu principal” contém ao menos 3 elementos do tipo “Link” e, provavelmente, pode ter também um rótulo. A figura 16 exemplifica como poderia ser esse elemento abstratamente.

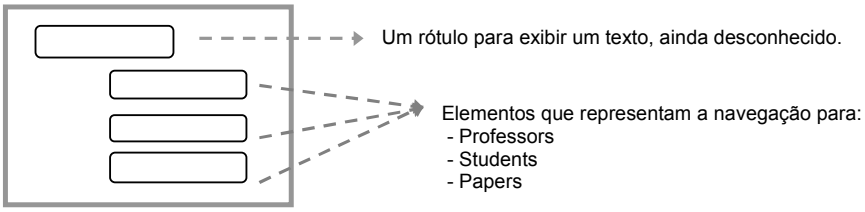

figura 16 - Representação abstrata do elemento “Main Menu”.

Tem-se um elemento de busca na área 2 da figura 15. Por meio do diagrama de contexto, sabe-se que este elemento pode realizar pesquisas sobre professores (índice “Professors from search”), alunos (índice “Students from search”) e artigos (índice “Papers from search”). Entretanto, não se sabe como esses elementos serão apresentados, pois é possível utilizar, por exemplo, um elemento do tipo “*RadioButton*” ou “*CheckBox*” para representar esses elementos. Além desses elementos, ele pode conter também um rótulo ou um outro elemento, ainda desconhecido. Um exemplo de como poderia ser esse elemento abstratamente é ilustrado na .

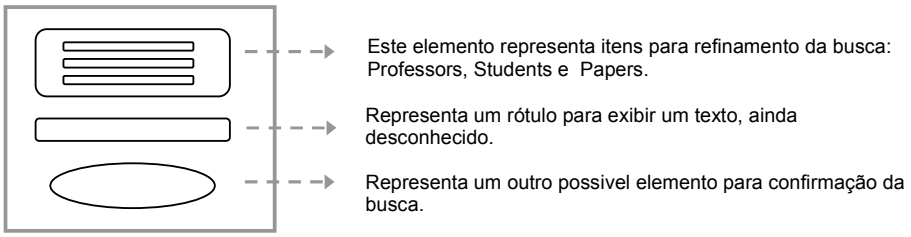

figura 17 – Representação abstrata do elemento “Search”.

Na área 3 da figura 15, tem-se um elemento que representa os dados de um professor específico. Por meio do diagrama de contexto, nota-se que este elemento permite uma navegação para os alunos relacionados; logo, pode ter um conjunto de “SimpleActivator”s. Provavelmente, ele terá um rótulo ou um conjunto de rótulos que irá descrever a instância de um professor específico; e como representa um contexto em ordem alfabética de nomes dos professores, poderá ter uma navegação interna entre professores. Na figura 18, pode-se visualizar um exemplo desse elemento abstratamente.

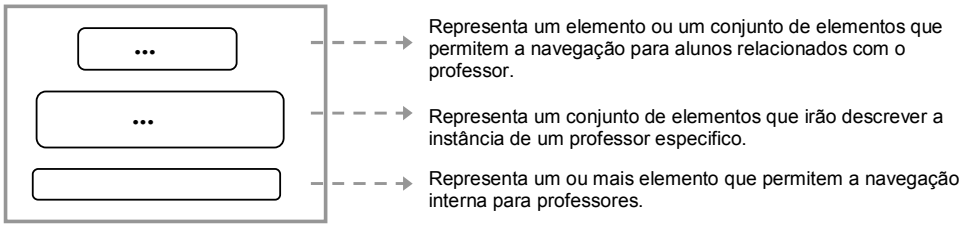

figura 18 – Representação abstrata do elemento que descreve dados de um professor.

Seguindo a lógica descrita acima, pode-se criar um exemplo de uma interface concreta composta por um menu principal, uma operação de busca e uma área contendo dados de um professor específico. Pode-se adicionar, por exemplo, um outro elemento que represente o título principal da página. Uma possível interface concreta é ilustrada na figura 19.

figura 19 – Interface Concreta.

Para gerar a interface concreta ilustrada pela figura 19 deve-se, primeiramente, realizar a modelagem abstrata dessa interface. A modelagem é uma descrição de todos os elementos que compõem a interface, baseada na ontologia de *Widgets* Abstratos. Quando a modelagem de uma interface abstrata é realizada, cria-se uma instância dessa ontologia. Portando, essa instância descreve, abstratamente, todos os elementos que compõem a interface.

Na figura 20 estão destacadas as composições dos elementos que constituem a interface da figura 19. Na figura 21 apresenta-se a instância da ontologia de *Widgets* Abstratos, representando abstratamente a interface concreta da figura 19 e demonstrando como são feitas as composições dos elementos da interface abstrata.

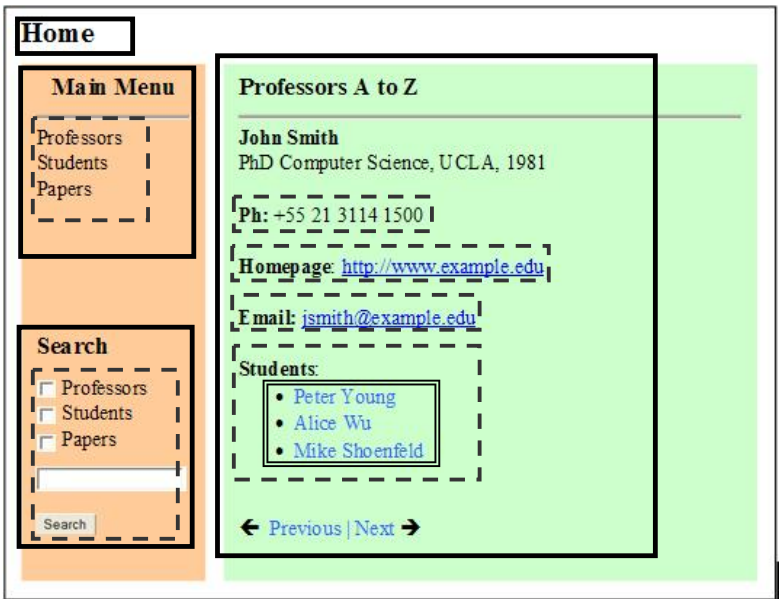

figura 20 - Composições dos elementos da interface concreta.

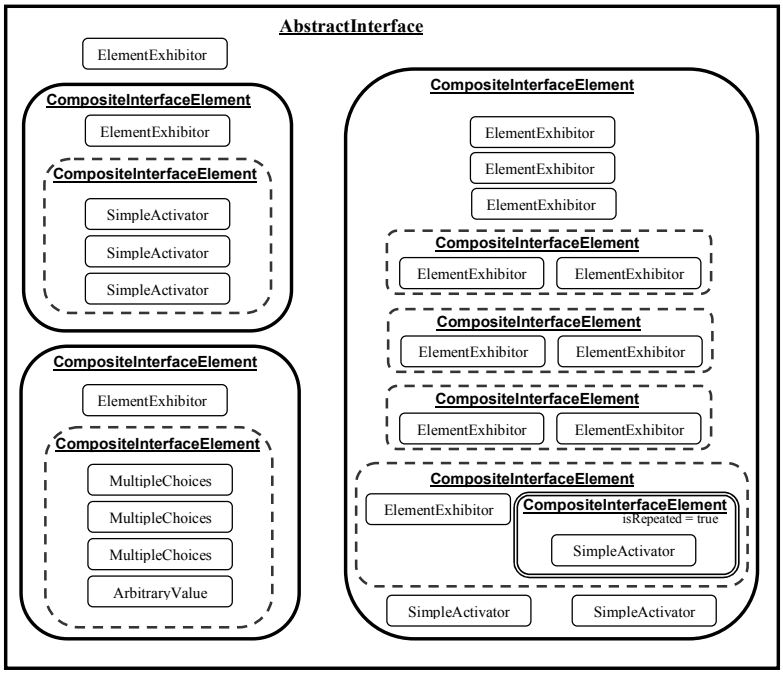

figura 21 – Representação abstrata das composições da interface concreta (figura 20).

A figura 21 representa a estrutura dos elementos que compõem a interface concreta da figura 19. No decorrer desta seção serão descritas as quatro principais partes (elementos) que compõem essa interface, detalhando a sua modelagem. Para detalhar a modelagem dos elementos de interface foi utilizada a notação N3 ao invés de OWL, pois N3 descreve os elementos e as suas propriedades de uma maneira mais legível se comparado com XML. A modelagem completa em N3 referente a esses elementos está descrita no Anexo A.

4.1.2.1. Título Principal

A descrição em notação N3 desse elemento pode ser visualizada na figura 22.

```
:HomeTitle
    a          awo:ElementExhibitor;
    awo:visualizationText "Home".
```

figura 22 - Modelagem abstrata do elemento “Título Principal”.

O elemento na figura 22 é do tipo “ElementExhibitor” e é composto apenas pela propriedade “visualizationText”. Essa propriedade recebe uma *string*, que será exibida por esse elemento. Neste caso, o elemento irá exibir a *string* “Home” na interface concreta.

4.1.2.2. Menu Principal

O “Menu Principal” da interface concreta é composto por dois elementos principais, sendo um do tipo “ElementExhibitor”, que representa a *string* “Main Menu”, e o outro do tipo “CompositeInterfaceElement”, que descreve uma composição de elementos. Na figura 23 pode-se visualizar graficamente a estrutura da instância abstrata do elemento “Main Menu”.

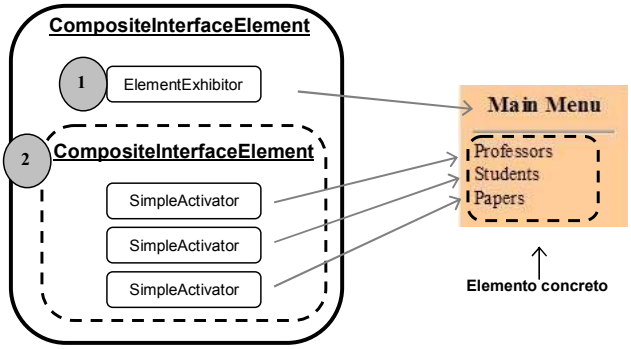

figura 23 – Estrutura abstrato do elemento “Main Menu”.

O elemento número 1 descrito na figura 23 é do tipo “ElementExhibitor”. Aqui ele representa uma *string* e a sua definição é composta apenas pela propriedade “visualizationText”. A descrição em notação N3 desse elemento é ilustrada na figura 24.

```
:IdxMainMenuTitle
    a   awo:ElementExhibitor;
    awo:visualizationText "Main Menu".
```

figura 24 – Modelagem abstrata do atributo *título* do elemento “Main Menu”.

O elemento número 2 descrito na figura 23, representa uma composição de elementos abstratos que definem os itens do “Main Menu”. Ele é composto por três elementos do tipo “SimpleActivator”. A definição em N3 dessa composição é apresentada na figura 25.

```
1  :MainMenuEntries
3      a     awo:CompositeInterfaceElement;
4      awo:hasInterfaceElement
5         :MainMenuProfessors, :MainMenuStudents, :MainMenuPapers;
6      awo:isRepeated "false".
7
8   :MainMenuProfessors
9     a      awo:SimpleActivator;
10     awo:fromAttribute "section";
11     awo:fromElement "MainMenuProfessors";
12     awo:targetInterface "Professors".
13
14  :MainMenuStudents
15      a   awo:SimpleActivator;
16     awo:fromAttribute "section";
17     awo:fromElement "MainMenuStudents";
18     awo:targetInterface "Students".
19
20 :MainMenuPapers
21     a    awo:SimpleActivator;
22     awo:fromAttribute "section";
23     awo:fromElement "MainMenuPapers";
24     awo:targetInterface "Papers".
```

figura 25 – Modelagem abstrata dos itens que compõem o “Main Menu”.

Apresenta-se na figura 25 a definição dos itens: “MainMenuProfessor” (linhas 8 a 12), “MainMenuStudents” (linhas 14 a 18) e “MainMenuPaper” (linhas 20 a 24). Todos esses itens possuem as mesmas propriedades que são: “targetInterface”, “fromElement” e “fromAttribute”.

A propriedade “targetInterface” (linhas 12, 18 e 24) indica qual será a instância abstrata que será criada quando o elemento for ativado. As propriedades “fromElement” (linhas 11, 17 e 23) e “fromAttribute” (linhas 10, 16 e 22) indicam qual o elemento e o atributo navegacional (da instância da ontologia navegacional do modelo SHDM), a que esse elemento abstrato se refere.

O elemento que representa a composição de todos os elementos descritos acima é o “MainMenuEntries” (linhas 1 a 6) do tipo “CompositeInterfaceElement”. Ele tem as propriedades “isRepeated” e “hasInterfaceElement”. A propriedade “isRepeated” indica que os elementos pertencentes a essa composição podem se repetir ou não; neste caso eles não irão se repetir, pois o valor dessa propriedade é *false* (linha 6). A propriedade “hasInterfaceElement” (linha 4) indica quais são os elementos que fazem parte dessa composição, que neste caso são três: “MainMenuProfessor”, “MainMenuStudents” e “MainMenuPaper”, descrito na linha 5.

4.1.2.3. Operação de Busca

O elemento que representa a operação de busca na interface concreta é composto por dois elementos principais, um do tipo "ElementExhibitor" e outro do tipo "CompositeInterfaceElement". A *string* "Search" é representada pelo elemento do tipo "ElementExhibitor"; já o "CompositeInterfaceElement" representa uma composição de elementos abstratos, os quais definem as opções para a realização da operação de busca. A estrutura da instância abstrata desse elemento é ilustrada na figura 26.

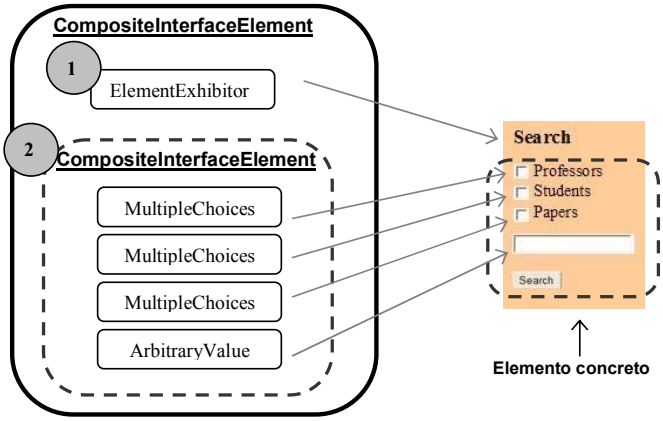

figura 26 - Estrutura abstrata do elemento de busca.

O elemento número 1 da figura 26 é do tipo "ElementExhibitor" e sua definição em N3 pode ser visualizada na figura 27. A descrição desse elemento é semelhante a definição do mesmo na figura 24.

```
:TitleSearch
    a       awo:ElementExhibitor ;
    awo:visualizationText  "Search" .
```

figura 27 - Modelagem abstrata do atributo *título* do elemento de *busca*.

O elemento número 2 da figura 26 representa uma composição de quatro elementos, três do tipo "MultipleChoices" e um do tipo "ArbitraryValue". A definição abstrata desses elementos em OWL é descrita na figura 28.

```
:1  SearchElements
 2      a       awo:CompositeInterfaceElement ;
 3      awo:fromIndex "idxSearch" ;
 4      awo:hasInterfaceElement
 5           :SearchProfessors, :SearchStudents, :SearchPapers, :SearchField;
 6      awo:isRepeated "false" ;
 7      awo:targetInterface "SearchResult" .
8
9   :SearchProfessors
10     a       awo:MultipleChoices ;
11     awo:fromAttribute "section" ;
12     awo:fromElement "SearchProfessors" .
13
14  :SearchStudents
15     a       awo:MultipleChoices ;
16     awo:fromAttribute "section" ;
17     awo:fromElement "SearchProfessors" .
18
19  :SearchPapers
20     a       awo:MultipleChoices ;
21     awo:fromAttribute "section" ;
22     awo:fromElement "SearchProfessors" .
23
24  :SearchField
25     a       awo:IndefiniteVariable .
```

figura 28 – Modelagem abstrata dos itens que compõem o elemento de busca.

Na figura 28, os elementos “SearchProfessors” (linhas 9 a 12), “SearchStudents” (linhas 14 a 17) e “SearchPapers” (linhas 19 a 22) são do tipo “MultipleChoices” e possuem as mesmas propriedades: “fromElement” e “fromAttribute”. Essas duas propriedades representam respectivamente o elemento e o atributo navegacional referente a esses elementos, na instância da ontologia navegacional dessa aplicação.

O elemento “SearchField” descrito na linha 24 da figura 28 não possui nenhuma propriedade, visto que esse elemento não irá exibir nenhuma informação. A sua função nesse contexto é receber informações (entrada) do usuário.

4.1.2.4. Dados de um Professor Específico

O elemento que descreve os dados de um professor na interface concreta, representa uma composição de nove elementos principais, três do tipo “ElementExhibitor”, dois do tipo “SimpleActivator” e quatro do tipo “CompositeInterfaceElement”. A estrutura da instância abstrata dessa composição é ilustrada na figura 29.

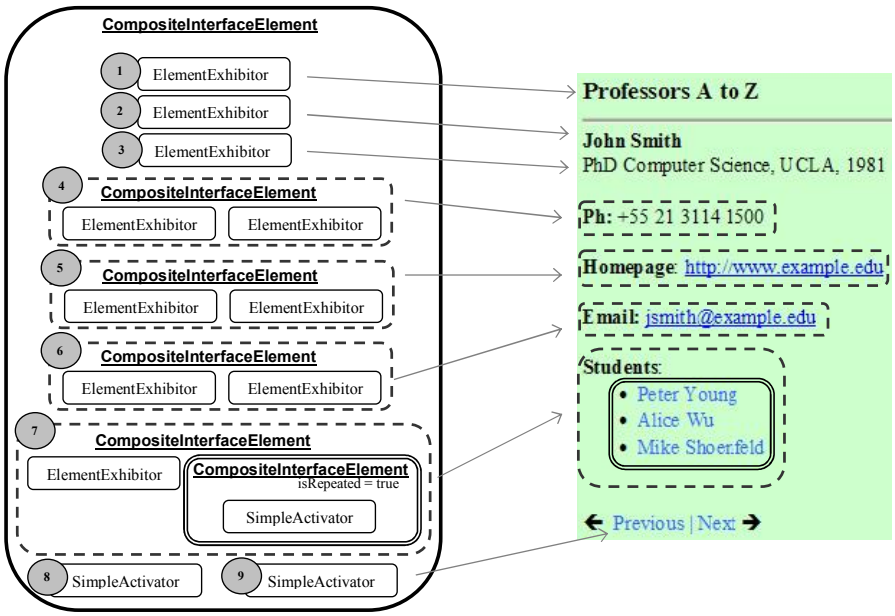

figura 29 – Estrutura abstrata do elemento que representa os dados do professor.

Os elementos de número 1, 2 e 3 da figura 29 são do tipo "ElementExhibitor". Eles representam respectivamente o título, o nome do professor e a sua formação, na interface concreta. A definição abstrata desses elementos em N3 é descrita na figura 30.

```
1  :IdxProfessorAlphTitle
2      a      awo:ElementExhibitor ;
3      awo:visualizationText   "Professors from A to Z" .
4
5  :ProfessorName
6      a      awo:ElementExhibitor ;
7      awo:fromAttribute "name" .
8
9  ProfessorDegree
10     a      awo:ElementExhibitor ;
11     awo:fromAttribute "degree" .
```

figura 30 – Modelagem abstrata dos elementos de número 1, 2 e 3 da figura 29.

Na figura 30 os elementos "ProfessorName" (linhas 5 a 7) e "ProfessorDegree" (linhas 9 a 11) são do tipo "ElementExhibitor" e possuem a mesma propriedade: "fromAttribute". Essa propriedade representa o atributo navegacional referente a esses elementos, na instância da ontologia navegacional dessa aplicação. Esse atributo navegacional contém o texto a ser exibido por cada um desses elementos na interface concreta. Esse texto é obtido pela aplicação a partir de um Java Bean que contém o valor de cada atributo.

O elemento "idxProfessorAlphaTitle" (linhas 1 a 3) também é do tipo "ElementExhibitor", mas possui a propriedade "VisualizationText". Essa propriedade já contém o texto a ser exibido por esse elemento.

As definições das composições de número 4, 5 e 6 da figura 29, que representam, respectivamente, os dados sobre o “telefone”, “homepage” e o “e-mail” do professor, são semelhantes. Cada uma dessas composições possui dois elementos do tipo ”ElementExhibitor”. Portanto, será apresentada a definição em N3 de apenas uma dessas composições. A definição que representa a composição do telefone do professor é descrita na figura abaixo.

```
1   :ProfessorPhoneComposition
3       a       awo:CompositeInterfaceElement ;
4       awo:fromContext "ctxProfessorAlpha" ;
5       awo:hasInterfaceElement
6               : ProfessorPhoneLabel , :ProfessorPhone ;
7       awo:isRepeated "false" .
8
9   :ProfessorPhoneLabel
10      a       awo:ElementExhibitor ;
11      awo:visualizationText "Ph:" .
12
13  :ProfessorPhone
14      a       awo:ElementExhibitor ;
15      awo:fromAttribute "phone" .
```

figura 31 - Modelagem abstrata do atributo *telefone* do elemento *professor*.

O elemento “ProfessorPhoneComposition” da figura 31 é composto por três propriedades: “fromContext”, “isRepeated” e “hasInterfaceElement”. A propriedade “fromContext” indica qual o contexto a que este elemento pertence. A propriedade “isRepeated” indica que os elementos pertencentes a essa composição podem se repetir ou não; neste caso eles não irão se repetir, pois o valor dessa propriedade é *false* (linha 7).

A propriedade “hasInterfaceElement” indica quais são os elementos que fazem parte dessa composição, que neste caso são dois: “ProfessorPhoneLabel” e “ProfessorPhone”. O elemento “ProfessorPhoneLabel” (linhas 9 a 11) é do tipo “ElementExhibitor” e possui a propriedade “visualizationText”, que recebe como valor a *string* “Ph:”. O elemento “ProfessorPhone” também é do tipo “ElementExhibitor” (linhas 13 a 15), mas possui a propriedade “fromAttribute”, que já foi explicada anteriormente. Esta propriedade é utilizada para obter a *string* que será exibida por este elemento na interface concreta, através do Java Bean que estará disponível no arquivo JSP.

O elemento número 7 da figura 29 representa a lista dos estudantes relacionados a um professor. Esse elemento representa uma composição, que difere um pouco das composições vistas anteriormente, pois esta possui um

elemento que pode ser repetido quantas vezes forem necessárias. A definição em N3 desse elemento está descrita na figura abaixo.

```
1  :StudentsByProfComposition
2       a       awo:CompositeInterfaceElement ;
3       awo:fromContext "ctxProfessorAlpha" ;
4       awo:hasInterfaceElement
5               :StudentByProfessorAlphaTitle , :StudentsByProfessorAlphaCompos ;
6       awo:isRepeated "false" .
7
8  :StudentByProfessorAlphaTitle
9       a       awo:ElementExhibitor ;
10      awo:visualizationText "Students:" .
11
12 :StudentsByProfessorAlphaCompos
13      a       awo:CompositeInterfaceElement ;
14      awo:fromAttribute "students" ;
15      awo:fromElement "ctxProfessorAlpha" ;
16      awo:hasInterfaceElement
17              :IdxStudentByProfessorAlpha ;
18    awo:isRepeated "true" .
19
20 :IdxStudentByProfessorAlpha
21      a       awo:SimpleActivator ;
22      awo:fromAttribute "name" ;
23      awo:targetInterface "StudentByProfessor" .
```

figura 32 – Modelagem abstrata da lista de estudantes de um professor.

O elemento “StudentByProfComposition” (linhas 1 a 6) da figura 32 é composto por três propriedades: “fromContext”, “isRepeated” e “hasInterfaceElement”, que já foram explicadas anteriormente. A partir da propriedade “hasInterfaceElement”, sabe-se que esta composição possui dois elementos: “StudentByProfessorAlphaTitle” e “StudentByprofessorAlphaCompos”. O elemento “StudentByProfessorAlpha Title” (linhas 8 a 10) é do tipo “ElementExhibitor” e possui a propriedade “visualizationText” cujo valor é a *string* “Students:”, que será exibida por esse elemento.

O elemento “StudentByprofessorAlphaCompos” (linhas 12 a 18), que representa a lista de nomes dos estudantes, é uma outra composição que possui quatro propriedades: “fromElement”, “fromAttribute”, “isRepeated” e “hasInterfaceElement”. As propriedades “fromElement” e “fromAttribute”, indicam respectivamente o elemento e o atributo navegacional referente e esse elemento, na instância da ontologia navegacional dessa aplicação. A propriedade “isRepeated” indica que os elementos pertencentes a essa composição poderão ou não se repetir. Neste caso, o valor dessa propriedade é *true* (linha 18), indicando que os elementos dessa composição poderão se repetir.

A partir da propriedade “hasInterfaceElement” desse elemento, sabe-se que esta composição possui apenas um elemento: “idxStudentByprofessorAlpha”. Este elemento é que representa o nome de cada um dos estudantes e é ele que pode se repetir quantas vezes forem necessárias. Ele é do tipo “SimpleActivator” e possui duas propriedades: “fomAttribute” e “targetInterface”. A propriedade “targetInterface” indica qual será a interface abstrata a ser criada quando esse elemento for ativado; e a propriedade “fromAttribute” já foi explicada anteriormente.

Os elementos número 8 e 9 da figura 29 representam respectivamente os links “Next” e “Previous” da interface concreta. A definição abstrata em N3 desses dois elementos está descrita na figura 33.

```
1  :ProfessorAlphaNext
2      a      awo:SimpleActivator ;
3      awo:fromAttribute "_NEXT" ;
4      awo:fromElement "ctxProfessorAlpha" ;
5      awo:targetInterface "ProfessorAlpha" .
6
7  :ProfessorAlphaPrevious
8      a      awo:SimpleActivator ;
9      awo:fromAttribute "_PREV" ;
10     awo:fromElement "ctxProfessorAlpha" ;
11     awo:targetInterface "ProfessorAlpha" .
```

figura 33 – Modelagem abstrata dos elementos “Next” e “Previous”.

Na figura 33 os elementos “ProfessorAlphaNext” (linhas 1 a 5) e “ProfessorAlphaPrevious” linhas (7 a 11) são do tipo “SimpleActivator” e possuem as mesmas propriedades: “fromElement”, “fromAttribute” e “targetInterface”,que já foram explicadas anteriormente.

4.2. Mapeamento para o Elemento Concreto

Após o desenvolvimento da interface com os seus elementos abstratos deve-se indicar qual o elemento concreto que corresponde a cada elemento abstrato definido nessa interface. Com esse objetivo, cada elemento abstrato deverá ser mapeado em uma descrição que representa um elemento concreto. Para essa descrição pode-se utilizar qualquer uma das linguagens descritas no Capítulo 3 (XUL, Laszlo, XAML, dentre outras), visto que elas realizam uma descrição dos elementos muito mais concreta que a proposta dessa dissertação. Dessa maneira, pode-se afirmar que essas linguagens estão mais direcionadas para a ontologia de *Widgets* Concretos do que para a de *Widgets* Abstratos.

Define-se então uma outra ontologia que descreve elementos concretos para facilitar o processo de mapeamento (ontologia de *Widgets* Concretos). Assim, pode-se tornar a interface abstrata definida independentemente de tecnologia e, além disso, estabelecer regras de consistência para cada elemento abstrato indicando quais os elementos concretos que ele poderá ser mapeado. Esses elementos concretos, na verdade, representam uma classe de *widgets* concretos; ou seja, quando se mapeia um elemento abstrato para um elemento concreto do tipo "*RadioButton*", esse "*RadioButton*" pode ser da linguagem XUL, XAML, HTML ou de outra linguagem que define elementos de interface.

Apresentar-se a seguir a ontologia de *Widgets* Concretos em detalhes e exemplos de mapeamento de elemento abstrato em elemento concreto.

4.2.1. Ontologia de *Widgets* Concretos

Essa ontologia descreve os possíveis elementos concretos de interface das aplicações hipermídia, denominados *widgets* concretos. Essas descrições são consideradas simples, posto que não descrevem muitos detalhes dos *widgets*. Os elementos descritos nessa ontologia são os mais utilizados atualmente nas aplicações; no entanto, na URL http://www.xulplanet.com é possível encontrar outros elementos não tratados por esse trabalho.

A ontologia de *Widgets* Abstratos utiliza essa mesma ontologia para especificar qual o elemento concreto que o seu elemento abstrato poderá ser mapeado. Esse mapeamento é necessário para a geração da interface concreta, que é realizada por uma aplicação que interpreta uma instância da ontologia de *widgets* abstrata, que representa uma interface abstrata. Essa instância possui vários elementos abstratos, onde cada um deles é mapeado em um elemento concreto.

Até o presente, a ontologia de *Widgets* Concretos, possui os elementos (instâncias) descritos na figura 34:

```
<ConcreteInterfaceElement rdf:ID="Button" />
<ConcreteInterfaceElement rdf:ID="CheckBox" />
<ConcreteInterfaceElement rdf:ID="CheckBoxAction" />
<ConcreteInterfaceElement rdf:ID="ComboBox" />
<ConcreteInterfaceElement rdf:ID="ComboBoxAction" />
<ConcreteInterfaceElement rdf:ID="ComboBoxTarget" />
<ConcreteInterfaceElement rdf:ID="Composition" />
<ConcreteInterfaceElement rdf:ID="Form" />
<ConcreteInterfaceElement rdf:ID="Image" />
<ConcreteInterfaceElement rdf:ID="Label" />
<ConcreteInterfaceElement rdf:ID="Link" />
<ConcreteInterfaceElement rdf:ID="RadioButon" />
<ConcreteInterfaceElement rdf:ID="RadioButonAction" />
<ConcreteInterfaceElement rdf:ID="RadioButonTarget" />
<ConcreteInterfaceElement rdf:ID="TextArea" />
<ConcreteInterfaceElement rdf:ID="TextBox" />
```

figura 34 – Instâncias que representam os elementos concretos.

A seguir serão descritos os elemento ilustrados na figura 34, que representam instâncias da ontologia de *Widgets* Concretos,.

- *Button*: representa os elementos que possuem funções embutidas a serem realizadas como: *submit* e *reset*. Essas funções são executadas quando esse elemento é ativado, através do *mouse* ou do teclado;
- *CheckBox*: representa um tipo de botão que possui 2 estados: selecionado ou não selecionado. O usuário pode alterar o estado desse elemento, clicando com o *mouse* ou via teclado, na caixa de verificação desse elemento. Muitas vezes são utilizados grupos desse mesmo elemento, representando uma lista de opções, onde podem ser selecionados *n* elementos. Esse elemento é composto de um *label* e de um botão (caixa de verificação).
- *CheckBoxAction*: representa um elemento do tipo *form*, composto de dois elementos distintos: um *CheckBox* (descrito anteriormente) e um *Button* (com a ação de *submit*);
- *ComboBox*: representa uma lista de itens. Consiste em uma caixa de entrada de texto e de um menu, a partir do qual o usuário pode selecionar uma dentre as diversas alternativas (uma lista);
- *ComboBoxAction*: representa um elemento do tipo *form*, composto de dois elementos distintos: um *ComboBox* (descrito anteriormente) e um *Button* (com a ação de *submit*);
- *ComboBoxTarget*: representa o elemento *ComboBox* (descrito anteriormente), composto de uma lista de elementos, onde cada elemento representa um *Link* específico, ou seja, quando o usuário

realizar a escolha do elemento e selecioná-lo na lista, automaticamente será executada uma ação, podendo ser esta a chamada de uma interface abstrata;

- *Composite*: representa composições de elementos de interface. Uma interface é mapeada em um elemento concreto *composite*, pois ela representa uma composição de vários elementos de interface. Podem-se mapear outras composições mais simples de elementos para esse conceito;
- *Form*: representa um conjunto de elementos de interface. Ele possui dois atributos: *method* (*get* ou *post* - método *http* para a submissão do formulário) e *action* (contém uma informação, indicando o que vai acontecer quando o formulário for submetido). Essa *action* pode conter um endereço *http*, um nome de uma página, um endereço eletrônico, entre outras informações. Quando o botão de um elemento *form* é selecionado, todos os valores dos elementos concretos que o compõem, são submetidos para a URL descrita no atributo *action*. O código HTML desse elemento é composto da descrição de um botão com a função de *submit*;
- *Image*: representa elementos concretos que exibem figuras;
- *Label*: conhecido como rótulo, representa um valor (texto/string) que é exibido na interface;
- *Link*: representa ligações pelas quais se pode navegar para outra parte:
    - do mesmo documento (outra parte na mesma página);
    - de outro documento (página, arquivo de imagem) da mesma aplicação hipermídia;
    - de outro documento, em qualquer computador da rede;
    - e, evidentemente, para se copiar todos esses arquivos (download)
- *RadioButton*: representa um tipo de botão que possui 2 estados: selecionado ou não selecionado. Seu estado pode ser alterado pelo clique do *mouse* ou via teclado na caixa de verificação desse elemento. Na maioria das vezes são utilizados grupos desse

elemento para representar um conjunto de opções, onde é permitida a seleção de apenas um elemento do conjunto;

- *RadioButtonAction*: representa um elemento do tipo *form* composto de dois elementos distintos: um *RadioButton* (descrito anteriormente) e um *Button* (com a ação de *submit*);
- *RadioButtonTarget*: representa o elemento *RadioButton* (descrito anteriormente) com um *Link* embutido na definição de cada item da lista desse elemento;
- *TextBox*: representa um campo de entrada de informação. Este campo é composto de apenas uma linha e pode ter *n* colunas;
- *TextArea*: representa um campo de entrada de informação. Esse campo é composto de *n* linha e *n* colunas e ele pode conter barras de rolagem horizontal de vertical, se for o caso.

4.2.2. Mapeamento

Deve ser realizado um mapeamento de cada elemento abstrato, definido em uma instancia de interface abstrata (ontologia de *Widgets* Abstratos), para um elemento concreto definido na ontologia de *Widgets* Concretos.

Deve-se seguir algumas regras para realizar esse mapeamento, visto que não é permitido o mapeamento de um elemento abstrato para qualquer elemento concreto. Existem regras de consistência especificando os elementos concretos nos quais cada elemento abstrato pode ser mapeado.

Apresenta-se a seguir o exemplo de uma regra de consistência de um elemento abstrato, neste caso “SimpleActivator”. Essa regra é definida na propriedade “mapsTo” da ontologia de *Widgets* Abstratos e todos os conceitos dessa ontologia possuem essa propriedade.

```
...
@prefix cwo:    <http://www.tecweb.inf.puc-rio.br//ontology/CW/cwo#> .
@prefix awo:    <http://www.tecweb.inf.puc-rio.br/ontology/AW/awo#> .
...

:SimpleActivator
    a       owl:Class ;
    rdfs:subClassOf awo:AbstractInterfaceElement ;
    rdfs:subClassOf
          [ a       owl:Restriction ;
            owl:allValuesFrom
                  [ a       owl:Class ;
                    owl:oneOf (cwo:Link cwo:Button)
                  ] ;
            owl:onProperty awo:mapsTo
          ] .

cwo:Link
    a       cwo:ConcreteInterfaceElem .

cwo:Button
    a       cwo:ConcreteInterfaceElem .
```

Figura 35 – Regra de mapeamennto

Cada elemento, definido na ontologia de *Widgets* Abstrato, possui uma regra específica. Essa regra define os elementos concretos, descritos na ontologia de *Widgets* Concretos, em que o elemento abstrato pode ser mapeado. No exemplo, percebe-se que o elemento do tipo "SimpleActivator", só poderá ser mapeado para o elemento concreto do tipo "*Link*" ou "*Button*"da ontologia de *Widgets* Abstratos

No decorrer desta seção será apresentado o mapeamento de todos os elementos que compõem a interface abstrata ilustrada na figura 21.

4.2.2.1. Título Principal

O título principal da interface é mapeado para o elemento concreto do tipo *Label*. O seu mapeamento é realizado pela propriedade "mapsTo" e está descrito na Figura 36.

```
:HomeTitle
    a       awo:ElementExhibitor ;
    awo:mapsTo cwo:Label .
```

Figura 36 - Mapeamento do elemento abstrato: "título principal da página".

4.2.2.2. Menu Principal

Como visto anteriormente, esse elemento é composto por dois elementos principais: um representando o título do menu e o outro uma composição de itens.

O elemento que representa o título desse menu é mapeado para o elemento concreto do tipo *Label,* seu mapeamento é descrito, em notação N3, na Figura 37.

```
:IdxMainMenuTitle
    a       awo:ElementExhibitor ;
    awo:mapsTo cwo:Label .
```

Figura 37 - Mapeamento do elemento abstrato: título do “Main Menu”.

A composição dos itens do menu principal possui três elementos que são mapeados para o elemento concreto do tipo *Link*. A Figura 38 apresenta o mapeamento dessa composição e de cada um dos seus elementos em notação N3.

```
:MainMenuEntries
    a       awo:CompositeInterfaceElement ;
    awo:mapsTo cwo:Composition .

:MainMenuProfessors
    a       awo:SimpleActivator ;
    awo:mapsTo cwo:Link .

:MainMenuStudents
    a       awo:SimpleActivator ;
    awo:mapsTo cwo:Link .

:MainMenuPapers
    a       awo:SimpleActivator ;
    awo:mapsTo cwo:Link .
```

Figura 38 – Mapeamento dos elementos abstrato: itens do “Main Menu”.

4.2.2.3. Operação de Busca

O elemento que representa a operação de busca na interface concreta é composto por dois elementos principais, um do tipo “ElementExhibitor” e o outro do tipo “CompositeInterfaceElement”. O mapeamento do elemento “ElementExhibitor”, que representa o título dessa operação de busca, é descrito na Figura 39.

```
:TitleSearch
    a       awo:ElementExhibitor ;
    awo:mapsTo cwo:Label .
```

Figura 39 – Mapeamento do elemento abstrato: “título do elemento Search”.

O mapeamento do elemento “CompositeInterfaceElement”, que representa a composição das opções de busca dessa operação, é ilustrado na Figura 40.

```
:SearchElements
    a       awo:CompositeInterfaceElement ;
    awo:mapsTo cwo:Form .

:SearchProfessors
    a       awo:MultipleChoices ;
    awo:mapsTo cwo:CheckBox .

:SearchStudents
    a       awo:MultipleChoices ;
    awo:mapsTo cwo:CheckBox

:SearchPapers
    a       awo:MultipleChoices ;
    awo:mapsTo cwo:CheckBox .

:SearchField
    a       awo:IndefiniteVariable ;
    awo:mapsTo cwo:TextBox .
```

Figura 40 – Mapeamento dos elementos abstratos: “itens do elemento Search”.

4.2.2.4. Dados de um Professor Específico

O elemento que descreve os dados de um professor na interface concreta, descrito anteriormente, representa uma composição de nove elementos principais, três do tipo “ElementExhibitor”, dois do tipo “SimpleActivator” e quatro do tipo “CompositeInterfaceElement”.

Na figura 29 os elementos de número 1 (linhas 1 a 3 da Figura 41), 2 (linhas 5 a 7 da Figura 41) e 3 (linhas 9 a 11 da Figura 41) são todos mapeados para o elemento concreto do tipo *Label*, Esses mapeamentos são descritos na Figura 41.

```
1   :IdxProfessorAlphTitle
2       a       awo:ElementExhibitor ;
3       awo:mapsTo cwo:Label .
4
5   :ProfessorName
6       a       awo:ElementExhibitor ;
7       awo:mapsTo cwo:Label .
8
9   :ProfessorDegree
10      a       awo:ElementExhibitor ;
11      awo:mapsTo cwo:Label .
```

Figura 41 – Mapeamento dos elementos abstratos de número 1, 2 e 3 da figura 29.

Os elementos de número 4, 5 e 6 da figura 29 são composições de dois elementos cada um. Suas definições são semelhantes e são mapeados para os mesmos elementos. Por esse motivo a Figura 42 ilustra apenas o mapeamento do elemento composto que representa os dados do telefone do professor.

```
1  :ProfessorPhoneComposition
2      a      awo:CompositeInterfaceElement ;
3      awo:mapsTo cwo:Composition .
4
5  :ProfessorPhoneLabel
6      a      awo:ElementExhibitor ;
7      awo:mapsTo cwo:Label .
8
9  :ProfessorPhone
10     a      awo:ElementExhibitor ;
11     awo:mapsTo cwo:Label .
```

Figura 42 – Mapeamento do elemento abstrato: “dados do telefone do professor”.

A Figura 42 mostra o mapeamento do elemento composto “ProfessorPhoneComposition” (linha 3) para um elemento concreto do tipo *Composition* e o mapeamento dos seus elementos para um elemento concreto do tipo *Label* (linha 7 e 11).

O elemento número 7 da figura 29 é uma composição de dois elementos principais, um do tipo “ElementExhibitor” e o outro do tipo “CompositionInterfaceElement”. O mapeamento do “ElementExhibitor” é descrito na Figura 43.

```
1  :StudentByProfessorAlphaLabel
2      a      awo:ElementExhibitor ;
3      awo:mapsTo cwo:Label .
```

Figura 43 – Mapeamento do elemento abstrato: “título da lista de estudantes”.

O mapeamento do elemento “StudentByProfessorAlphaComposition”, que representa uma composição de nomes dos estudantes, é ilustrado na Figura 44.

```
1  :StudentsByProfessorAlphaComposition
2      a      awo:CompositeInterfaceElement ;
3      awo:compositionTag "<li>" ;
4      awo:mapsTo cwo:Composition .
5
6  :IdxStudentByProfessorAlpha
7      a      awo:SimpleActivator ;
8      awo:mapsTo cwo:Link .
```

Figura 44 – Mapeamento do elemento abstrato: “lista de estudantes do professor”.

Na Figura 44, para realizar o mapeamento do elemento “StudentByProfessorAlphaCompositon”, que é uma composição, foram utilizadas duas propriedades, “mapsTo” e “compositonTag”. A propriedade “mapsTo” indica qual o elemento concreto que esse elemento abstrato será mapeado; já a “compositionTag” indica qual a *tag* HTML que irá separar os elementos que compõem essa composição. Esta última propriedade é necessária, pois na definição da modelagem abstrata desse elemento, descrita na figura 32, (linhas 12 a 18) a propriedade “isRepeated” recebeu o valor *true*. Desta forma, sabe-se que

os elementos que pertencem a essa composição poderão se repetir, portando é necessário, indicar qual a *tag* HTML que irá separar cada elemento da repetição. Neste caso, cada elemento é do tipo *link*, visto que o elemento, "ïdxStudentByProfessorAlpha" que pertence a essa composição, é mapeado para o elemento concreto do tipo "*Link"*.

Os elementos de número 8 e 9 da figura 29 são do tipo "SimpleActivator" e são mapeados para o elemento concreto do tipo *Link*. O mapeamento desses elementos é descrito na Figura 45.

```
:ProfessorAlphaNext
    a       awo:SimpleActivator ;
    awo:mapsTo cwo:Link .

:ProfessorAlphaPrevious
    a       awo:SimpleActivator ;
    awo:mapsTo cwo:Link .
```

Figura 45 – Mapeamento dos elementos abstratos: "Next" e "Previous".

4.3. Definição de layout dos elementos abstratos

Após ter se definido a interface abstrata e realizado o mapeamento de seus elementos, pode-se então definir um *layout* para a interface. A proposta desta dissertação utiliza o modelo de caixas ("box model") do padrão CSS, indicando *tags* HTML e classes do CSS que definirão uma "caixa" CSS para cada elemento descrito na interface abstrata. A definição de um layout para um elemento é definida a partir da propriedade "blockElement", cujo valor é precisamente o elemento HTML que define a "caixa" CSS.

No decorrer desta seção será apresentada a definição de alguns *layouts* para alguns dos elementos de interface abstrata modelados anteriormente.

4.3.1. Definição de *layout*

A Figura 46 ilustra o exemplo de um *layout* definido para o elemento abstrato que representa o "Título principal" da interface concreta. Esse *layout* é composto apenas pela tag HTML "<h1>"

```
: HomeTitle
    a       awo:ElementExhibitor ;
    awo:blockElement "<h1>" .
```

Figura 46 – Definição do *layout* para o elemento abstrato "Titulo Principal"

Na Figura 47 é descrito o exemplo de um *layout* definido para o elemento abstrato que representa o título do "Main Menu" da interface concreta. O *layout* deste elemento é composto de uma tag HTML (div) e de uma classe do CSS (class="titleMenu").

```
:IdxMainMenuTitle
    a       awo:ElementExhibitor ;
    awo:blockElement "<div class= titleMenu>" .
```

Figura 47 – Definição do *layout* para o elemento abstrato: "título do Main Menu".

Na Figura 48 apresenta-se o exemplo de um *layout* definido para o elemento abstrato, que representa os itens do "Main Menu".

```
:MainMenuEntries
    a       awo:CompositeInterfaceElement ;
    awo:blockElement "<div class='menu'>" .
```

Figura 48 - Definição do *layout* para elemento abstrato: "itens do Main Menu".

Desta forma na definição de um *layout*, para qualquer elemento abstrato, é necessário apenas definir *tags* HTML e classes CSS na propriedade "blockElement" do elemento abstrato. A integração dos elementos do CSS ao HTML será feita na geração da página, que será apresentada no Capítulo 5.

4.4. Exemplo

Aqui se apresenta um exemplo da modelagem completa, em notação N3, de um elemento da interface abstrata já modelado. A figura abaixo apresenta a modelagem, o mapeamento e a definição de um *layout* para o elemento abstrato "MainMenu".

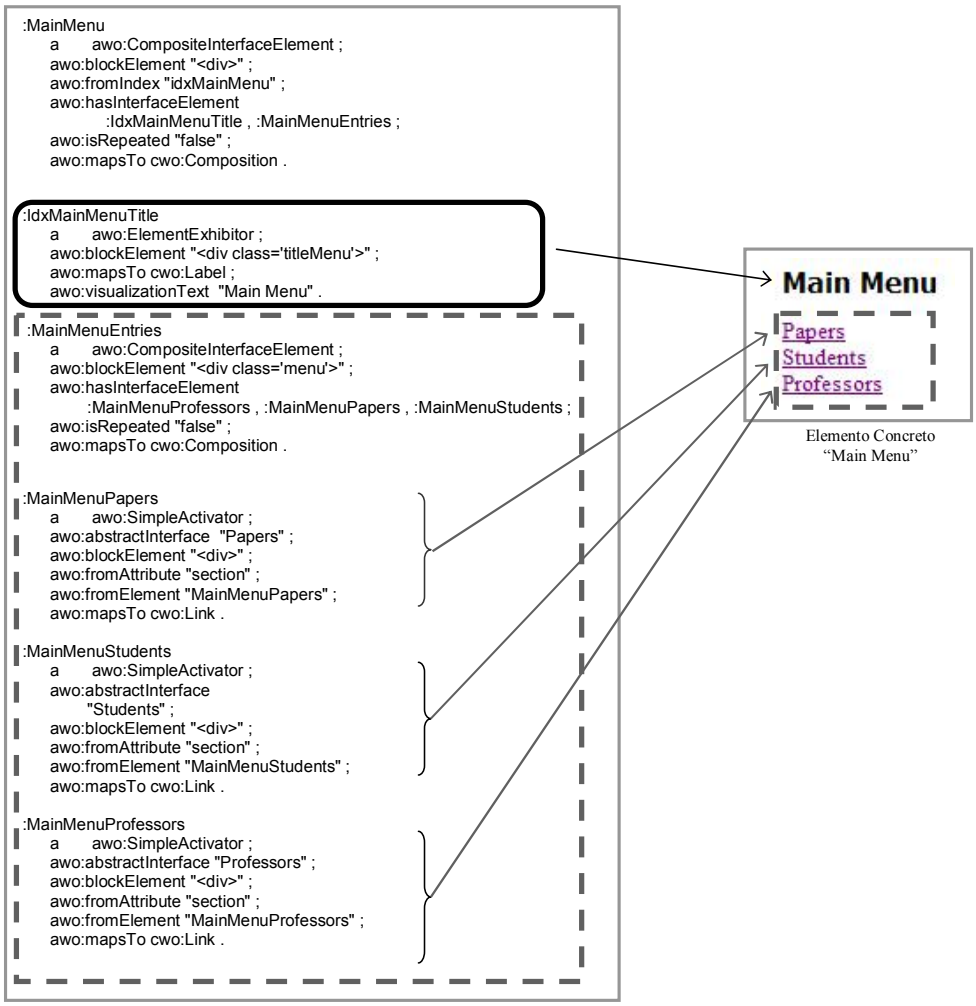

Figura 49 - Modelagem abstrata Completa do elemento “Main Menu”.