# 3
# Navegação Facetada

A classificação da informação em facetas foi inicialmente proposta por S.R.Ranganathan, em 1930 (Ranganathan, 1962). Tal conceito foi inicialmente aplicado na categorização da informação em bibliotecas e livrarias. Atualmente, vem se tornado bastante popular, principalmente na área de buscas na Web, por ser um mecanismo auxiliar aos mecanismos de busca até então desenvolvidos. Alguns modelos de navegação em facetas já estão disponíveis na Web, tais como: eBayExpress.com, bomdefaro.com.

Um sistema de recuperação de informação é caracterizado como sendo facetado se ele possui um conjunto mínimo de operações que permitem o usuário navegar de forma arbitrária em um conjunto de elementos, progressivamente, restringindo-o a um subconjunto mínimo definido pela relação de interseção entre as facetas (Suominen et al., 2007).

Nosso objetivo neste capítulo é desenvolver um modelo de navegação facetada para o Explorator. Para tanto, analisaremos diversas plataformas existentes no intuito de capturar estruturas que são intrínsecas desse paradigma. Como resultado, desenvolveremos uma linguagem para especificação de facetas, um modelo de navegação facetada baseado no nosso modelo de operações e um algoritmo para geração automática de facetas.

## 3.1.
## O que é uma faceta?

Faceta é um conjunto de termo/valores que são utilizados como filtros de um conjunto alvo. Um termo representa uma propriedade do conjunto de elementos sendo facetados e seus valores representam os possíveis valores que essa propriedade pode assumir. Os valores de uma faceta podem ser representados em uma taxonomia de valores, por exemplo: Brasil – Rio de Janeiro – Niterói.

A figura abaixo mostra um conjunto de facetas extraídas da interface do sistema Flamenco[38].

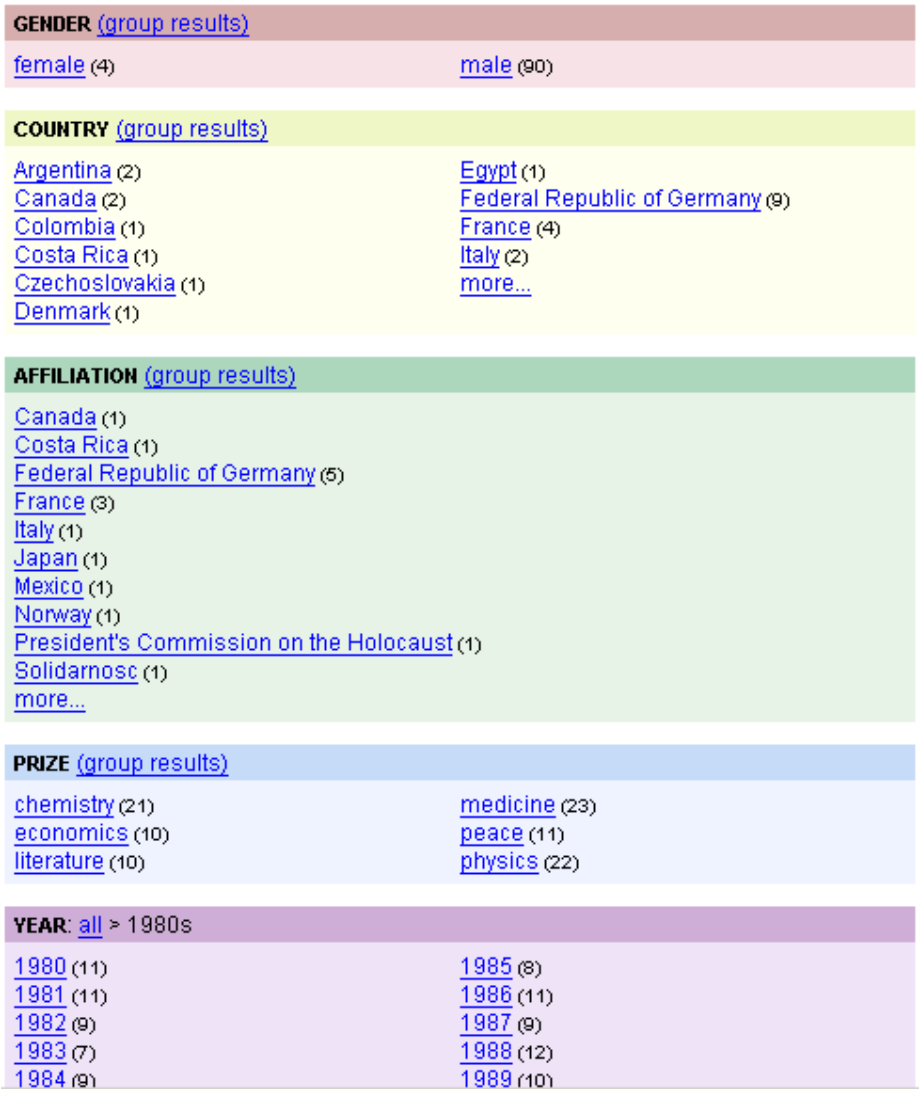


Figura 7 – Exemplo da interface facetada do Flamenco

## 3.2. Sistemas existentes

Abaixo faremos um breve sumário sobre os sistemas existentes no intuito de captar características comuns entre eles.

[38] http://flamenco.berkeley.edu/index.html

### 3.2.1. Flamenco

Flamenco é um dos primeiros sistemas Web de navegação facetada. O foco do Flamenco é prover uma interface de navegação para grandes coleções de dados, de forma que o usuário possa navegar sem se sentir perdido.

Como modelo de representação das facetas, o Flamenco possui um conjunto de arquivos textos que devem ser preenchidos seguindo uma determinada convenção. Tais arquivos são processados e os dados são adicionados em tabelas de um banco de dados relacional. Tais arquivos nos permitem definir termos e hierarquia entre eles.

O Flamenco não é baseado no modelo RDF e todas as configurações das facetas devem ser providas manualmente pelo engenheiro do sistema.


Figura 8 – Interface do Flamenco

### 3.2.2. BrowseRDF

BrowseRDF[39] é um sistema Web de navegação em facetas sobre dados RDF. Todas as facetas são geradas automaticamente, extraídas da base RDF sendo navegada. O projetista não é capaz de personalizar as facetas de nenhuma maneira. O foco de tal ferramenta é permitir uma navegação facetada para dados RDF, sendo que as facetas são geradas automaticamente.

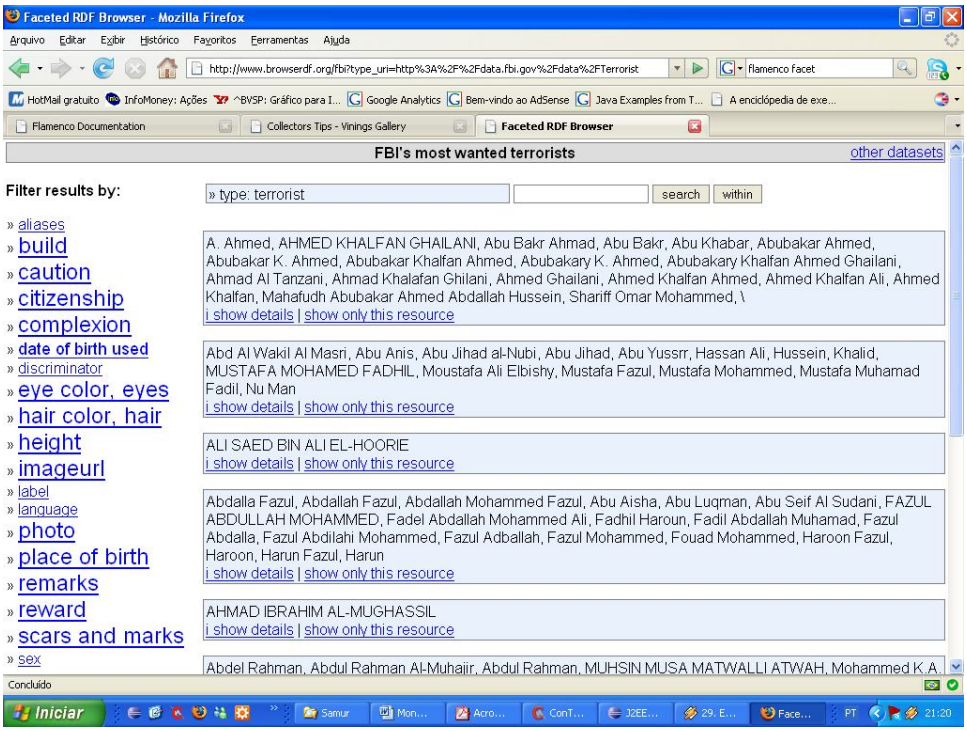


Figura 9 – Interface do BrowseRDF

### 3.2.3. FacetMap

FacetMap[40] é um sistema Web de navegação em facetas e usa um modelo próprio para definição das facetas e armazenamento dos dados. As facetas são

[39] http://browserdf.org/

[40] www.facetmap.com

expressas através da linguagem XFML[41], que é uma linguagem para definição de hierarquia de termos ou facetas.

O FacetMap permite a personalização da interface através de folhas de estilo XSLT para definição do layout da interface.

O FacetMap não suporta o modelo RDF.


Figura 10 – Interface do FacetMap

### 3.2.4. Longwell

Longwell[42] é um sistema Web de navegação sobre dados RDF que utiliza o paradigma de navegação em facetas. Desenvolvido em Java, permite vários níveis de personalização da interface do usuário. Longwell permite a navegação direta sobre dados RDF, e também possui um modelo capaz de definir quais propriedades do grafo RDF comporão a hierarquia de facetas. Longwell suporta a meta-linguagem Fresnel (Bizer et al., 2006), que possibilita a seleção e formatação de um conjunto de triplas RDF. Tal mecanismo pode ser utilizado na

[41] http://www.xml.com/pub/a/2003/01/22/xfml.html

[42] http://simile.mit.edu/wiki/Longwell

personalização da interface. Longwell utiliza um formato proprietário para armazenamento dos dados, porém os arquivos de entrada são arquivo RDF/XML, N3 ou NT.

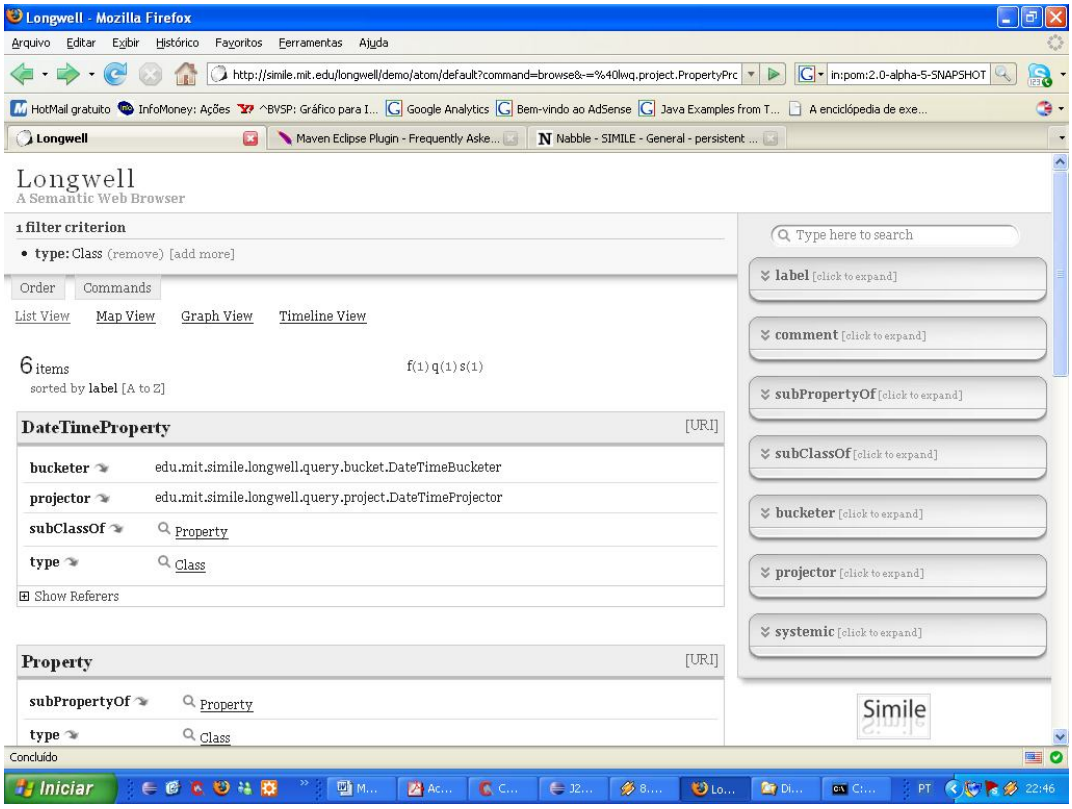


Figura 11 – Interface do Longwell

### 3.2.5. Avaliação dos Modelos Existentes

Basicamente, os sistemas de navegação facetada são compostos por: uma interface, um modelo de operações e uma linguagem de representação das facetas.

#### 3.2.5.1. Operações de Navegação Facetada

Na teoria, a navegação facetada (Hearst M., 2006) suporta apenas a operação de interseção. Começando com o conjunto de todos os elementos, a cada

passo é feita a interseção do conjunto corrente com o conjunto definido pela faceta selecionada.

No entanto, verificamos que alguns sistemas, tais como o Longwell, implementam um modelo de navegação mais abrangente, suportando também a operação de união. Neste caso, a união significa que o usuário pode selecionar na interface mais de um valor de uma faceta, formando assim um conjunto interseção entre a união dos conjuntos formados pela seleção de valores na faceta.

Já o BrowserRDF (Oren et al., 2006) propôs um modelo de navegação ainda mais abrangente. Tal modelo é mais amplo que o modelo de navegação facetada que estamos tratando aqui, por também definir operações de navegação entre os recursos. No fundo o que esse modelo prove é uma forma mais flexível de facetar conjuntos arbitrários, que não precisam ser explicitamente definidos pelo projetista do sistema, como pode ser visto na citação abaixo:

“Navegação facetada pode ser considerada simultaneamente como construir e caminhar por uma árvore de decisão, cujos ramos representam predicados e cujos nodos representam restrição aos valores. Por exemplo, fig. 6a (fig 12) mostra uma árvore para navegar em uma coleção de publicações, restringindo primeiramente pelo autor, depois o ano e, finalmente, o tópico...

Um caminho na árvore representa um conjunto de restrições que selecionam os recursos de interesse. A árvore é construída de forma dinâmica, por exemplo, os valores disponíveis para se restringir "tópico" são diferentes nas duas árvores: Fig. 6b mostra todos os tópicos de publicações em 2000, e a Fig. 6a mostra apenas tópicos de Stefan Decker.“ (Oren et al., 2006, p.8).

No entanto, se avaliamos cada conjunto individualmente, a operação suportada por esse modelo continua sendo a operação de interseção.

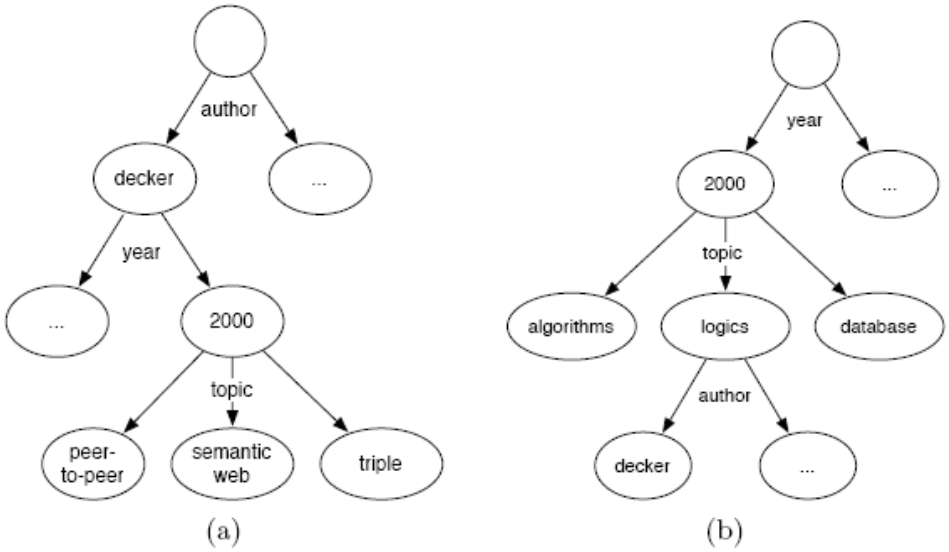


Figura 12 – Figura 6 referida na citação de (Oren et al., 2006, p.8)



### 3.2.5.2. Estruturas de Interface

Identificamos três grupos de informação existentes nas interfaces avaliadas: *recursos sendo facetados, facetas e facetas selecionadas.* Diversas são as formas que estas informações são apresentadas na interface. O importante aqui é identificar que elas existem.

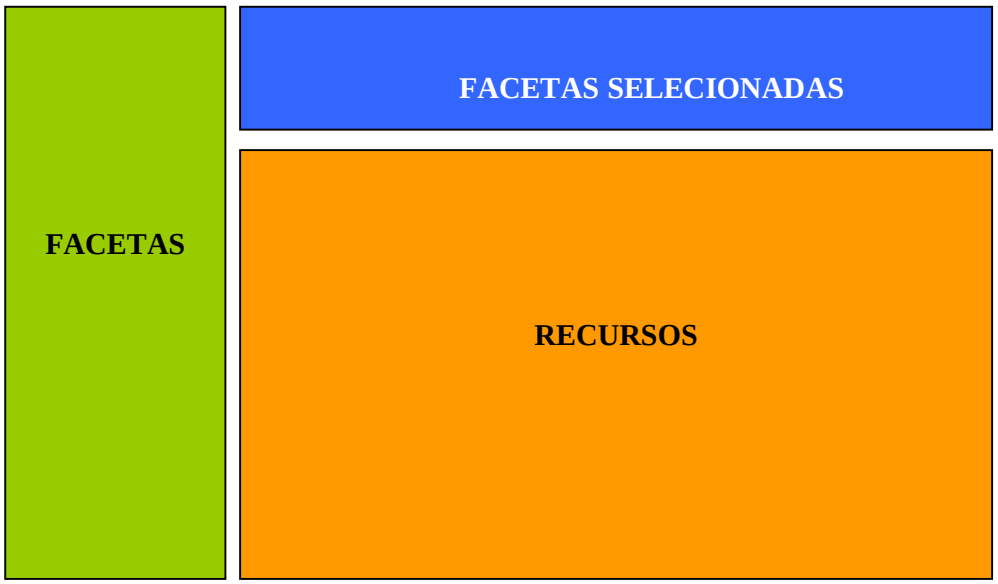


Figura 13 – Estruturas de uma interface de navegação facetada

### 3.2.5.2.1. Facetas

Facetas representam a hierarquia dos termos que são utilizados na classificação dos itens e na navegação do usuário. O usuário seleciona uma dessas facetas a cada passo de sua navegação e a cada seleção um conjunto de recursos classificados pela faceta é retornado. Em termos de interface, a principal função dessa área é disponibilizar ao usuário o conjunto de facetas para que ele possa filtrar o conjunto de recursos sendo facetado. Nas aplicações investigadas, essa área exibe um conjunto de termo/valores organizados em uma taxonomia.

Podemos identificar dois itens de informação nesta área:

- Faceta (termo/valores)
- Cardinalidade da Faceta – número de elementos que serão retornados, caso seja selecionada uma determinada faceta.

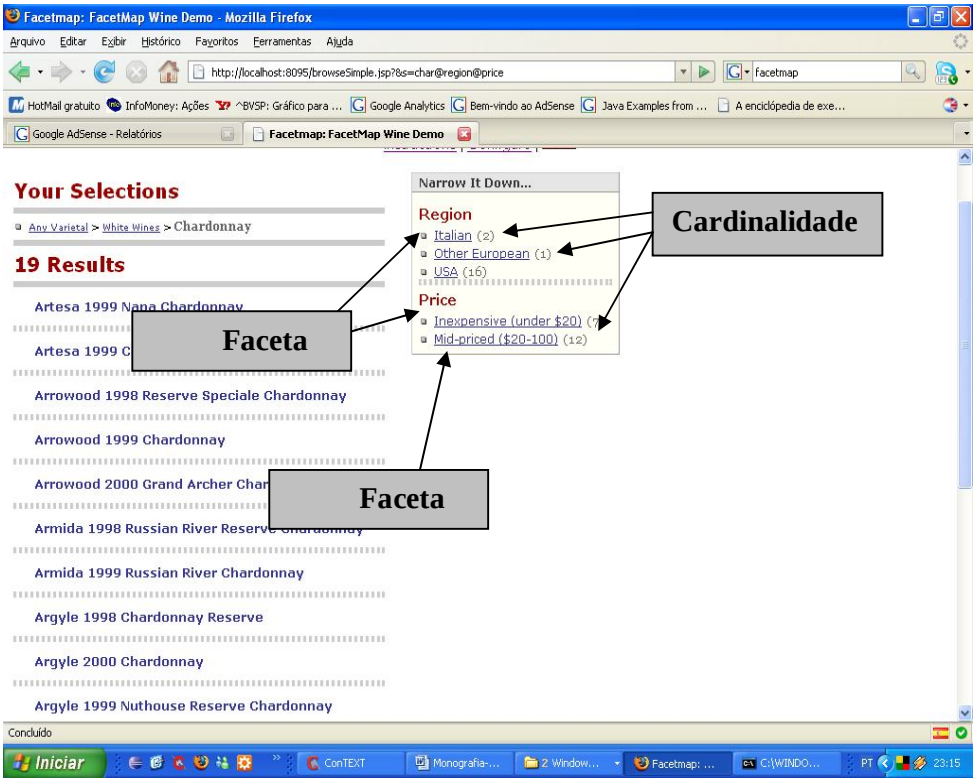


Figura 14 - Tela do FacetMap demonstrando as facetas e a cardinalidade.

#### 3.2.5.2.2. Facetas Selecionadas

Facetas Selecionadas é o conjunto de facetas selecionadas pelo usuário. Esse grupo de informação é bem evidente em qualquer interface facetada, pois é através dela que o usuário controla as facetas previamente selecionadas. Esse é um elemento importante na navegação facetada e existem diversas opções de representação na interface.

#### 3.2.5.2.3. Recursos

Recursos são os itens de informação que são filtrados pela seleção das facetas.

Abaixo podemos verificar os 3 componentes acima na interface do Flamenco.

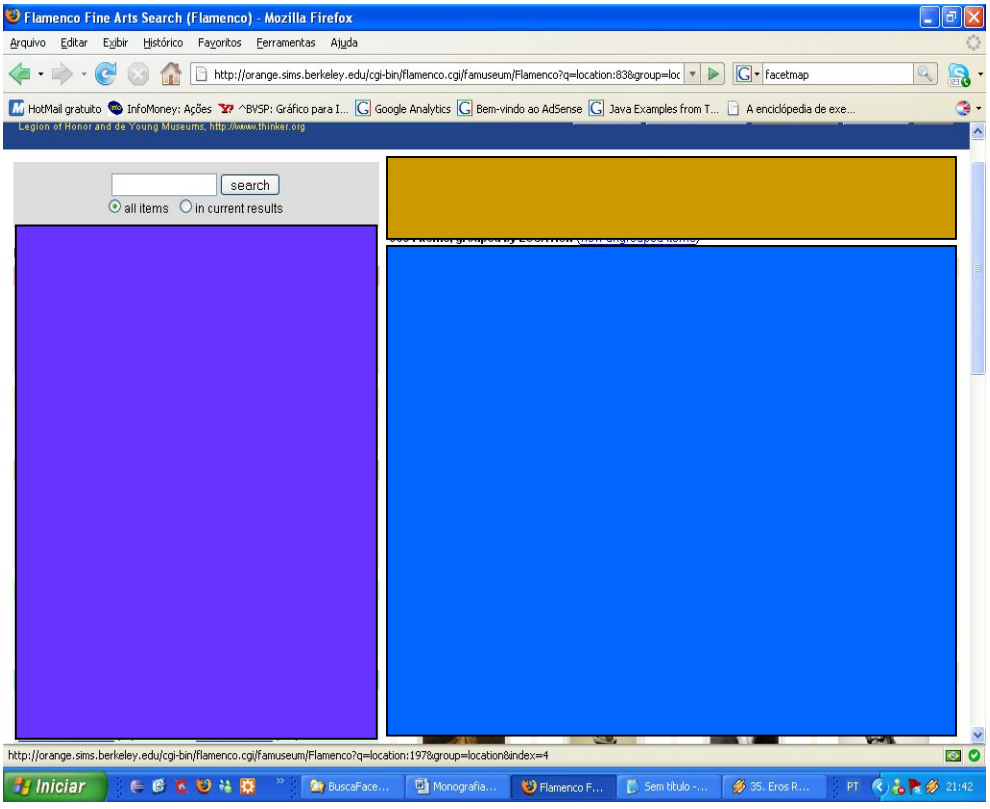


Figura 15 – Tela do Flamenco evidenciando os 3 grupos de informação de uma interface de navegação facetada.

### 3.2.5.3. Especificação das Facetas

A linguagem de especificação de facetas é o elemento mais importante do modelo, pois ela cria o elo entre os dados e a interface facetada. Tal linguagem define qual filtro deverá ser aplicado aos elementos facetados, uma vez que um valor da faceta for selecionado.

Cada sistema utiliza uma abordagem distinta para especificação das facetas. O FacetMap permite que utilizemos a linguagem XFML para descrição das facetas. Já o sistema Flamenco possui um formato de arquivo específico no qual podemos descrever a hierarquia das facetas, que é bastante similar a abordagem proposta pelo FacetMap. O BrowseRDF infere as facetas diretamente da base de dados RDF e não possui um modelo de representação explícito. Dito de outra forma, o modelo do BrowseRDF é o próprio RDF. Já o Longwell é um pouco mais flexível quanto à especificação das facetas, sendo capaz de inferir as facetas diretamente dos dados RDF e também possui um formato de arquivo proprietário, no qual o usuário pode descrever cada faceta do sistema. Longwell utiliza também, na especificação das facetas, a linguagem Fresnel, que é capaz de definir qual informação contida no grafo RDF será exibida ao usuário, além de definir características de sua formatação.

#### 3.2.5.3.1. XFML

A linguagem XFML é uma meta-linguagem para representação de hierarquia de termos, que pode ser utilizada na construção de uma estrutura facetada. Ela permite definirmos termos e a relação entre eles, além de permitir associarmos os dados aos termos. O trecho de código abaixo é um código XFML que descreve a hierarquia a seguir:

- City
  - New York
  - L.A.
- Type of place
  - Bars
  - Restaurants
- Type of music
  - Blues
  - Latin

Quadro 10 - Hierarquia utilizada para exemplificar a linguagem XFML

```
<facet id="city">City</facet>
<facet id="place">Type of place</facet>
<facet id="music">Type of music</facet>
<topic id="ny" facetid="city"><name>New York</name></topic>
<topic id="la" facetid="city"><name>Los Angeles</name></topic>
<topic id="bar" facetid="place"><name>bar</name></topic>
<topic id="restaurant" facetid="place"><name>restaurant</name></topic>
<topic id="blues" facetid="music"><name>blues</name></topic>
<topic id="latin" facetid="music"><name>latin</name></topic>
```

Quadro 11 – Exemplo do uso de XFML



Este paradigma proporciona a construção da hierarquia de facetas independentemente do modelo de dados. No entanto, os dados devem ser anotados para cada faceta, utilizando uma sintaxe específica da linguagem XFML, como mostra o exemplo abaixo:

```
<page url="http://bbkingblues.com/">
<title>B. B. Kind blues club and grill</title>
<description>Conveniently located in the heart of Times Square near Penn Station
and Port Authority, The B.B. King Blues Club and Grill offers music fans a unique
experi-ence. Owned by the Bensusan Family, proprietors of the world renowned
Blue Note Jazz Club, the club features world-class musical talent and consists of
two distinct spaces: the Showcase Room and Lucille's Grill.
</description>
 <occurrence topicid="bar" />
<occurrence topicid="blues" />
<occurrence topicid="ny" />
 </page>
```

Quadro 12 – Exemplo de uso de XFML, associando conteúdo com termos.

Além disso, a linguagem XFML é um bom modelo de representação, porém não é capaz de expressar valores computados, ou seja, valores determinados por uma função, como por exemplo, a concatenação entre os valores de duas

propriedades (ex.: altura e comprimento, formando dimensão). Isso limita consideravelmente a expressividade deste modelo.

#### 3.2.5.3.2. Fresnel

Fresnel é uma linguagem (vocabulário RDF) destinada a atingir dois objetivos na apresentação de dados RDF: especificar qual e como a informação contida no grafo RDF deverá ser apresentada.

O Longwell utiliza uma extensão da linguagem Fresnel para especificar facetas. A linguagem Fresnel possui um mecanismo similar ao XPath (*XML Path Language*)[43], que permite a seleção de elementos do grafo RDF possibilitando assim, que o usuário defina exatamente o que será apresentado e como.

Abaixo exemplificaremos o uso dessa linguagem. Se um conjunto de dados RDF utilizados no nosso exemplo:

```
:Chris rdf:type foaf:Person ;
    foaf:name "Chris Bizer" ;
    foaf:mbox <mailto:chris@bizer.de> ;
    foaf:mbox <mailto:bizer@gmx.de> ;
    foaf:depiction <http://www.wiwiss.fu-
berlin.de/suhl/ueber_uns/team/Fotos/ChrisPassbild.jpg> .
```

Quadro 13 – Dados RDF utilizados no exemplo da linguagem Fresnel.

O exemplo a seguir mostra como um grupo de formatações que são aplicadas a uma instância de *foaf:Person*.

[43] http://www.w3.org/TR/xpath

```
:foafGroup rdf:type fresnel:Group ;
        fresnel:stylesheetLink <http://www.example.org/example.css> ;
        fresnel:containerStyle "background-color: white;"^^fresnel:stylingInstructions ;

:foafPersonFormat rdf:type fresnel:Format ;
            fresnel:classFormatDomain foaf:Person ;
            fresnel:resourceStyle "background-color:
gray;"^^fresnel:stylingInstructions ;
            fresnel:group :foafGroup .

:nameFormat rdf:type fresnel:Format ;
         fresnel:propertyFormatDomain foaf:name ;
         fresnel:propertyStyle "border-top: solid black;"^^fresnel:stylingInstructions ;
         fresnel:labelStyle "font-weight: bold;"^^fresnel:stylingInstructions ;
         fresnel:group :foafGroup .

:urlFormat rdf:type fresnel:Format ;
        fresnel:propertyFormatDomain foaf:homepage ;
        fresnel:propertyFormatDomain foaf:mbox ;
        fresnel:value fresnel:externalLink ;
        fresnel:propertyStyle "border-top: solid black;"^^fresnel:stylingInstructions ;
        fresnel:labelStyle "font-weight: bold;"^^fresnel:stylingInstructions ;
        fresnel:group :foafGroup .

:depictFormat rdf:type fresnel:Format ;
          fresnel:propertyFormatDomain foaf:depiction ;
          fresnel:label fresnel:none ;
          fresnel:value fresnel:image ;
          fresnel:propertyStyle "border-top: solid black;"^^fresnel:stylingInstructions ;
          fresnel:group :foafGroup .
```

Quadro 14 – Exemplo de formatações definidas na linguagem Fresnel.

A figura a seguir representa o resultado da aplicação dessas formatações sobre o conjunto de dados RDF em questão.


Figura 16 – Resultado da aplicação das formatações Fresnel sobre os dados RDF.

#### 3.2.5.4. Conclusão Sobre a Avaliação

Verificamos que nenhum mecanismo existente possui um modelo de navegação facetada destinado ao modelo RDF. Nenhum deles suporta a navegação facetada em um *SPARQL Endpoint*. Os que suportam a navegação em dados RDF não possuem um modelo de especificação das facetas que possa ser portável. Acreditamos que um modelo baseado no próprio modelo RDF é o mais adequado.

Uma característica interessante foi observada no *BrowserRDF*, que possibilita a navegação facetada em conjuntos arbitrários. Tal característica não implica em aumentar a complexidade do modelo de navegação, mas sim possibilita a aplicação de uma mesma heurística de navegação em conjunto de elementos distintos.

Acreditamos que um modelo ideal de navegação facetada deve possuir um modelo bem definido de especificação das facetas e permitir a navegação em conjuntos arbitrários.

## 3.3. Casos de Uso

Iremos expor alguns cenários que nos ajudarão a entender as limitações das abordagens utilizadas acima, e também nos ajudará a identificar elementos que deverão ser considerados na linguagem de especificação de facetas para modelo de navegação facetada que iremos propor.

### 3.3.1. Cenário 1

Suponha que o projetista possua uma base de dados RDF de produtos com nome do produto e preço em Reais. Uma faceta desejada seria a faceta com o termo “Preço”, com valores “Barato” e “Caro”. No entanto, os preços são descritos como triplas RDF do tipo:

- Item1 preço R$2,00
- Item2 preço R$3,00
- Item3 preço R$82,00

Em tal situação, para descrevemos a faceta “Preço” com o valor “Barato”, na linguagem XFML teríamos que adicionar uma nova tripla para cada item na base RDF. Exemplo:

- Item1 preço “Barato”
- Item2 preço “Barato”
- Item3 preço “Caro”

Já utilizando a linguagem Fresnel seriamos capazes de definir uma regra em SPARQL para obter todos os preços menores que R$50,00 e nomeá-los como “Barato”.

A vantagem de se utilizar uma linguagem capaz de realizar consulta sobre o modelo RDF é que não precisamos replicar ou adicionar dados ao modelo RDF, para expressarmos o mesmo conteúdo com um formato diferente. Ou seja, uma faceta pode possuir valores computados dos valores existentes na base RDF.

### 3.3.2. Cenário 2

Suponha agora que estejamos especificando as facetas para uma base de dados RDF sobre celulares. Suponha que esta base RDF contenha as propriedades *screenwidth* e *screenheight*, que descrevem o tamanho da tela do dispositivo celular, ou seja, sua resolução. Sendo assim, o projetista decide criar a faceta

"Resolução" e formatá-la como sendo a concatenação *screenwidth* + 'x' + *screenheight* (128x128, 160x220, etc.).

Mais uma vez, para expressarmos a faceta "Resolução", sem termos que replicar dados na base, precisaremos utilizar uma linguagem que seja capaz de computar tais valores. A linguagem Fresnel é capaz de definir agrupamento de propriedades, porém não é capaz de expressar a operação de concatenação de valores. Note também, que uma vez selecionado o valor "128x160" da faceta "Resolução", o sistema deverá filtrar os recursos que tenham a propriedade *screenwidth*=128 e *screenheight*=160, e não filtrar a propriedade resolução = 128x160, dado que não existe tal valor na base. Para tanto, não só é necessário um modelo capaz de especificar a formatação do valor da faceta na interface, mas capaz de expressar como deverá ser montada a consulta ao ser selecionado um dos seus valores na interface. Nenhum sistema analisado acima é capaz de representar tal situação.

### 3.3.3. Cenário 3

Suponha que tenhamos um conjunto de triplas RDF que descrevem localidades. Por exemplo, suponha que tenhamos as triplas:

- Item1 fn:country "BRA"
- Item1 dp:country "Brasil"
- Item1 fn:region "Brasil"

Ao especificarmos as facetas do sistema, gostaríamos de definir a faceta "País" com o valor "Brasil". Para tanto, teríamos que indicar que as propriedades '*dp:country'*, '*fn:region*' e '*fn:country*' são iguais e que os valores 'Brasil' e 'BRA' também são iguais. Note que um arquivo RDF pode conter triplas anotadas com sintaxe distintas usando espaços de nome distintos para descreverem o mesmo item de informação. Dessa forma, é importante que a linguagem de especificação de faceta seja capaz de expressar tais similaridades entre os valores.

### 3.3.4. Cenário 4

Suponha que tenhamos uma base onde os recursos formem uma hierarquia de valores, por exemplo: País > Estado > Cidade. Muitas vezes queremos que uma faceta explicite essa relação nos seus valores, ou seja, que seus valores sejam representados dentro de uma hierarquia.

O sistema Flamenco é capaz de representar os valores de sua faceta em uma hierarquia de valores mas não possui uma linguagem para sua representação. Já o FacetMap também é capaz de expressar essa relação hierárquica e especificá-la de forma consistente no formato XFML

É fundamental que uma linguagem de especificação de facetas seja capaz de representar tal situação.



## 3.4. Modelo de Especificação das Facetas

Um modelo de especificação das facetas deve ser capaz de representar todos os cenários aqui descritos. Portanto, é importante que tal modelo reúna as principais características de cada sistema facetado que analisamos. Segue abaixo um quadro comparativo do que vimos até aqui:

| | **FacetMap** | **Longwell** | **BrowseRDF** | **Flamenco** |
|---|---|---|---|---|
| **Linguagem de Representação de Facetas** | XFML / Dados Hierarquizados | Fresnel | Não Há | Dados Hierarquizados |
| **Inferência das Facetas** | Não | Sim | Sim | Não |
| **Modelo de Dados** | Texto / XML | RDF | RDF | Base Relacional |
| **Ordenação das Facetas** | Definida pelo usuário | Algoritmo | Ordem Alfabética | Definida pelo Usuário |
| **Valores computados** | Não possui | Não possui | Não possui | Não possui |
| **Definição de sinônimo entre os valores** | Não possui | Não possui | Não possui | Não possui |

Tabela 1 – Lista de funcionalidades dos modelos de navegação facetada.

### 3.4.1. Visão Geral do Modelo

Podemos abstrair a representação de uma faceta em três camadas: Camada de Visualização, Camada Lógica e Camada do Modelo da Faceta.

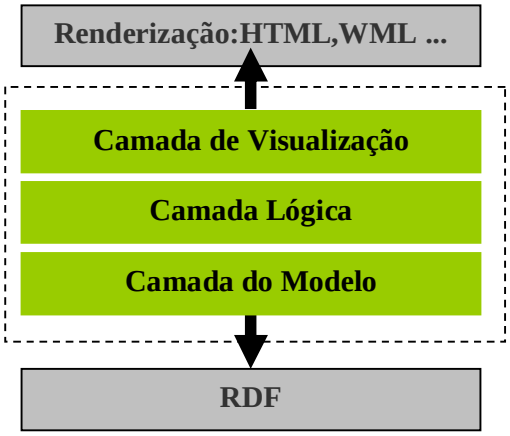


Figura 17 – Visão da arquitetura de um modelo de navegação facetada.

A camada de visualização representa o que o usuário visualizará na interface (Termo/Valores). A camada lógica representa as operações de ordenação que são aplicadas sobre as facetas, e as expressões que as definem. Já a camada do modelo representa o componente de consulta da faceta, ou seja, a expressão que define o filtro a ser aplicados aos recursos uma vez que uma faceta é selecionada.

A separação entre a camada lógica e do modelo se faz necessária, pois as facetas podem ser computadas em tempo de execução, e o valor da faceta na interface nem sempre existe realmente na base sendo filtrada. Em alguns casos, a valor da faceta na interface é exatamente o valor da sua propriedade no modelo de dados, e uma vez selecionada pelo usuário a operação a ser aplicada é apenas um filtro no conjunto de recursos que tenham aquela propriedade com aquele valor (termo/valor da faceta). Por exemplo, suponha que estejamos facetando um conjunto de pessoas, e para cada uma exista uma propriedade Local de Nascimento com os possíveis valores: RJ, SP, MG. Na especificação da facetas, os valores da facetas exibidos na interface (camada de visualização) são exatamente os valores disponíveis na propriedade Local de Nascimento; e ao selecionarmos ‘SP’ na interface, estaremos selecionando exatamente os recursos

que contenham a propriedade Local de Nascimento = ‘SP’. Por outro lado, podemos especificar uma faceta Resolução, que na interface seus valores são a concatenação das propriedades *width* e *height* (camada de visualização) existentes no modelo. Porém, ao selecionarmos uma das dimensões exibidas (128x128, 160x220, etc.), estaremos selecionando todos os dados que contenham a propriedades *width* e *heigth* iguais a um determinado valor, e não a propriedade Resolução em si, pois ela não existe no modelo de dados. Neste último caso, usamos a camada lógica para determinar o valor da faceta.

Uma abordagem alternativa é utilizar somente a camada do modelo da faceta, porém neste caso todos os possíveis valores das facetas devem existir ou serem adicionados na base RDF. Esta abordagem, apesar de mais simples, implica na replicação de dados na base RDF. Questões relacionadas à comparação da performance entre as duas abordagens estão fora do escopo desde trabalho, porém é valido que seja realizado um estudo futuro. Nesta dissertação, adoraremos a abordagem de usar as três camadas.



### 3.4.2. Características do Modelo

As seguintes características devem existir na linguagem de especificação das facetas:

1. O termo da faceta;
2. Os valores das facetas, inclusive computáveis;
3. A ordenação dos valores das facetas;
4. Operações de consulta sobre o modelo RDF;
5. Declaração de valores sinônimos.

### 3.4.3. Camada lógica

Como descrevemos, uma faceta é composta por um termo e um conjunto de valores que podem ser computados, organizados em uma hierarquia, etc.

O termo e valores da faceta podem ser especificados através da aplicação de uma consulta sobre os dados do modelo. Denominaremos esse tipo de faceta de faceta computada.

Note também, que em alguns sistemas tais como *Longwell* e *BrowseRDF* é possível inferir um conjunto de facetas diretamente do modelo RDF, e neste caso o termo e valores de um faceta denotam as propriedades RDF e seus valores.

Um conjunto de facetas pode estar ordenado pelos seus termos. Existem 3 abordagens de ordenação:

- Ordem alfabética;
- Ordem definida pelo usuário na especificação das facetas;
- Ordem definida por um algoritmo arbitrário.

Estas são informações de interface importantes que devem ser levadas em consideração na hora de se projetar uma linguagem de especificação de facetas

Outra característica contemplada dentro da camada lógica das facetas trata-se da relevância das facetas. A relevância de uma faceta indica o quanto uma faceta é capaz de discriminar os valores da base. Obviamente que, se ao selecionarmos uma faceta, ela não reduz o conjunto facetado, então não auxilia no processo de busca, e portanto não deve ser apresentada ao usuário. A faceta mais relevante é aquela capaz de discriminar o conjunto facetado em dois conjuntos com mesmo número de elementos. Sem levar em consideração qualquer feedback do usuário, as facetas mais relevantes devem ser apresentadas primeiramente aos usuários. Tal modelo de relevância é aplicável principalmente quando pretendemos gerar automaticamente as facetas dos dados RDF.



### 3.4.4. Camada do Modelo

Como descrevemos o modelo da faceta define expressões que selecionam dados na base uma vez que uma faceta é selecionada. Tais expressões podem ser visualizadas como uma consulta expressa em qualquer linguagem (SPARQL, SERQL, etc.) sobre a base RDF.

## 3.5. Vocabulário Faceto

Daremos o nome de FACETO ao vocabulário que utilizaremos para implementar uma linguagem de especificação de facetas. Utilizaremos o espaço

de nomes (*namespace*) *faceto*, para referenciarmos elementos desse vocabulário. A seguir estão descritas as estruturas desse vocabulário.

### 3.5.1. FacetGroup

As facetas são agrupadas em um **FacetGroup**. Um grupo de facetas pode ter um nome (*rdfs:label*) utilizado para identificar o grupo. Um grupo de facetas deve ter pelo menos uma faceta, que é identificada pela propriedade *faceto:facet*. Um grupo de facetas também pode ter um tipo, que serve para identificar se o grupo foi definido pelo usuário ou foi inferido automaticamente. A propriedade *faceto:type* é utilizada para descrever este valor.

```
@prefix rdfschema: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix faceto: <http://www.semanticnavigation.org/2008/faceto#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix dp1:  <http://sw.nokia.com/DP-1/> .
faceto:group1 rdf:type faceto:FacetGroup ;
        rdfschema:label "Group1" ;
          faceto:facet faceto:browser ;
          faceto:facet faceto:resolution ;
          faceto:facet faceto:length ;
          faceto:facet faceto:region .
          faceto:type faceto:userdefined .
```

Quadro 15 - Exemplo de um grupo de facetas

No exemplo acima existem 4 facetas definidas: *browser*, *resolution*, *length*, *region*. O grupo também possui a propriedade *faceto:type* com o valor *faceto:userdefined*, indicando que as facetas foram definidas pelo usuário. O outro valor possível para *faceto:type* seria *faceto:inferred*, indicando que tais facetas foram construídas automaticamente pelo sistema.

### 3.5.2. Facet

Cada faceta deve ser anotada como sendo do tipo *faceto:Facet*. Um *faceto:FacetGroup* pode conter várias *faceto:Facet*.

Toda faceta possui um termo e valores. Podemos definir o termo da faceta de duas formas distintas. A forma mais simples é definindo o valor do termo diretamente pela propriedade *faceto:literalTerm*. Neste caso, o termo será um valor *String* definido pela propriedade *faceto:literalTerm*. Veja o exemplo abaixo:

```
faceto:browser rdf:type faceto:Facet ;
        faceto:use dp1:browser ;
        faceto:literalTerm "Navegador" ;
```

Quadro 16 – Exemplo da definição de uma faceta com termo literal.

A outra forma de definirmos o termo é derivando seu valor de uma propriedade no meta-modelo de dados RDF. Neste caso, o valor do termo será o *rdfs:label* ou *localname* da propriedade que ele está sendo derivado. No exemplo abaixo, o valor do termo será a *rdfs:label* da propriedade *dp1:browser*. Usamos a propriedade *faceto:derivedTerm* para este tipo de definição.

```
faceto:browser rdf:type faceto:Facet ;
        faceto:use dp1:browser ;
        faceto:derivedTerm dp1:browser .
```

Quadro 17 – Exemplo da definição de um termo derivado de propriedade.

#### 3.5.2.1. Faceta com valores derivados de propriedades rdf

Os valores de uma faceta também podem ser definidos de formas variadas. A forma mais simples é derivar os valores de uma propriedade do meta-modelo de dados. Assim como na definição do termo, os valores derivados de propriedades, serão calculados baseados nos possíveis valores que a propriedade derivada pode assumir. Usamos a propriedades *faceto:use* para definir que os valores de uma faceta serão derivados de uma propriedade. Veja o exemplo abaixo:

```
faceto:browser rdf:type faceto:Facet ;
        faceto:use dp1:browser ;
        faceto:derivedTerm dp1:browser .
```

Quadro 18 – Exemplo de uma faceta derivada de um predicado RDF.

Também podemos usar a propriedade *faceto:useInverse* para indicar que os valores de uma faceta são derivadas de um propriedade inversa no domínio.

#### 3.5.2.2. Faceta com valores computados

Como descrevemos no Cenário 2 (vide 3.3.3.2) às vezes os valores de uma faceta precisam ser computados. Para definirmos uma faceta com valores computados usamos a propriedades *faceto:computedValue*. Uma faceta pode ter um ou mais valores computados. O valor computado de uma faceta deve ser um recurso com uma propriedade *faceto:expressionValue.* A expressão utilizada para computar o valor pode ser definida em qualquer linguagem. O Explorator possui implementação apenas para tratar expressões na linguagem Ruby. No exemplo abaixo a expressão *"dp1::screen_width + 'x' + dp1::screen_height"^^Ruby* denota que os valores serão uma concatenação de tais valores. Neste exemplo, estamos utilizando propriedades do meta-modelo de dados para computar os valores da faceta. Sendo assim, toda propriedade utilizada deve ser descrita com a propriedade faceto:use.

```
faceto:resolution rdf:type faceto:Facet ;
        faceto:literalTerm "Resolution" ;
        faceto:computedValue faceto:v1 ;
        faceto:use dp1:screen_height , dp1:screen_width .

faceto:v1 faceto:expressionValue "dp1::screen_width + 'x' + dp1::screen_height"^^Ruby ;
```

Quadro 19 – Exemplo de uma faceta computada.

Exemplo de faceta computada.

Quando os valores da faceta são computados, o cálculo da entropia da faceta incide sobre o valor computado e não em cada propriedade individualmente utilizada. Falaremos mais sobre a entropia de facetas no próximo capítulo.

#### 3.5.2.3. Faceta com valores escalares

O Cenário 1 (tópico 3.3.31) descreve uma situação recorrente na definição das facetas, que é quando os valores da faceta são valores escalares que representam intervalo de valores.

Para descrevermos que uma faceta possui um intervalo de valor, utilizamos a propriedade *faceto:computedValue*. Porém, o recurso computado deve possuir duas propriedades que não foram descritas até então: a propriedade *faceto:query* e *faceto:constraint.*

```
        faceto:length  rdf:type faceto:Facet ;
                faceto:computedValue faceto:v2 ;
                faceto:computedValue faceto:v3 ;
                faceto:derivedTerm dp1:length .

        faceto:v2 faceto:constraint "dp1::length.to_i < 100" ;
                 faceto:literalValue "Pequeno" ;
                 faceto:query  "filter('select{|resource|  resource.dp1::length.to_i  <  100}')
"^^Ruby.

        faceto:v3 faceto:constraint "dp1::length.to_i >100" ;
          faceto:literalValue "Grande" ;
          faceto:query     "filter('select{|resource|     resource.dp1::length.to_i     >100}')
"^^Ruby .
```

Quadro 20 – Exemplo de uma faceta com valores escalares.

Cada valor computado com a propriedade *faceto:constraint* define um único valor escalar. Tal valor somente aparecerá na lista de valores da faceta, caso a expressão *faceto:constraint* seja satisfeita para os recursos sendo facetados.

O valor da propriedade *faceto:query* deve ser uma expressão na linguagem de seleção implementada no sistema. Tal expressão será utilizada para filtrar os

elementos do conjunto sendo facetado. O valor computado com a propriedade *faceto:query* ignora qualquer propriedade *faceto:use* que tenha sido definida.

#### 3.5.2.4. Faceta com valores sinônimos

Muitas bases RDF possuem recursos diferentes que descrevem a mesma informação. Para evitar que uma faceta possua valores iguais que distinguem conceitualmente conjuntos iguais (vide Cenário 3, tópico 3.3.3.3) criamos o conceito de sinônimo. Através da propriedade *faceto:synonym* podemos definir que dois valores de uma faceta são iguais e, caso apareçam concomitantemente, ambos devem ser tratados como um só.

```
faceto:region  rdf:type faceto:Facet ;
            faceto:derivedTerm dp1:region ;
            faceto:use dp1:region .
            faceto:synonym faceto:s1 .
            faceto:synonym faceto:s2 .

faceto:s1 faceto:word dp1:Region/china ,
              <http://sw.nokia.com/FN-1/Country/CHN> ;
  faceto:defaultWord dp1:Region/china .

faceto:s2 faceto:word dp1:Region/europe ,
              <http://sw.nokia.com/FN-1/Region/europe> ;
  faceto:defaultWord dp1:Region/europe .
```

Quadro 21 - Exemplo da definição de uma faceta com sinônimo.

No exemplo acima, a faceta *Region* define sinônimos para a palavra “china” e “CHN”. Essa definição se faz necessária pois não queremos que a faceta *Region* possua dois valores distintos para representar o mesmo elemento conceitual no mundo real, neste caso o país China.

Um sinônimo também possui uma palavra padrão que será utilizada pela faceta na interface do usuário. Esse valor pode ser definido pela propriedade

*faceto:defaultWord* e deve ser um dos valores sinônimos definido pelas propriedades *faceto:word*.

### 3.5.2.5. Faceta com hierarquia de valores

O Faceto é capaz de representar uma hierarquia de valores em uma faceta. Para criarmos uma faceta com valores hierárquicos utilizamos um recurso com a propriedade *faceto:levelN*, onde N deve ser um valor inteiro de 1 a n. O valor para propriedade *facet:levelN* deve ser um recurso que define uma faceta.

```
        @prefix rdfschema: <http://www.w3.org/2000/01/rdf-schema#> .
        @prefix faceto: <http://www.semanticnavigation.org/2008/faceto#>
.
        @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
        @prefix modial:
<http://www.semWebtech.org/mondial/10/meta#/> .
        faceto:group1 rdf:type faceto:FacetGroup ;
                        rdfschema:label "Group1" ;
                          faceto:facet faceto:hieraquia ;
                          faceto:type faceto:userdefined .

        faceto: hieraquia rdf:type faceto:Facet ;
                         faceto:level1 faceto:country;
                           faceto:level2 faceto:province;

        faceto:country rdf:type faceto:Facet ;
                    faceto:computedValue faceto:v4.

        faceto:v4 faceto:queryValues "
Query.new.distinct(:s).where(:s,MONDIAL::hasProvince,:o).where(resour
ce, MONDIAL::cityIn, :o).execute" ;
                faceto:query
“Query.new.distinct(:s).where(resource,MONDIAL::hasProvince,:o).wher
e(:s, MONDIAL::cityIn, :o).execute”

        faceto:province       rdf:type faceto:Facet ;
                          faceto:use modial:cityIn ;
```

Quadro 22 – Exemplo de uma faceta com hierarquia de valores.

No exemplo acima, estamos facetando um conjunto de cidades; criamos uma faceta “hierarquia” que possui uma hierarquia de valores onde *faceto:country* aparece no primeiro nível e *faceto:province* no segundo nível da hierarquia. Na interface, a cada nível que vai sendo selecionado, o sistema passa a facetar o conjunto pelo próximo nível disponível na hierarquia.

### 3.5.3. Esquema Faceto

Abaixo seguem o diagrama do vocabulário faceto juntamente com a ontologia que o descreve.

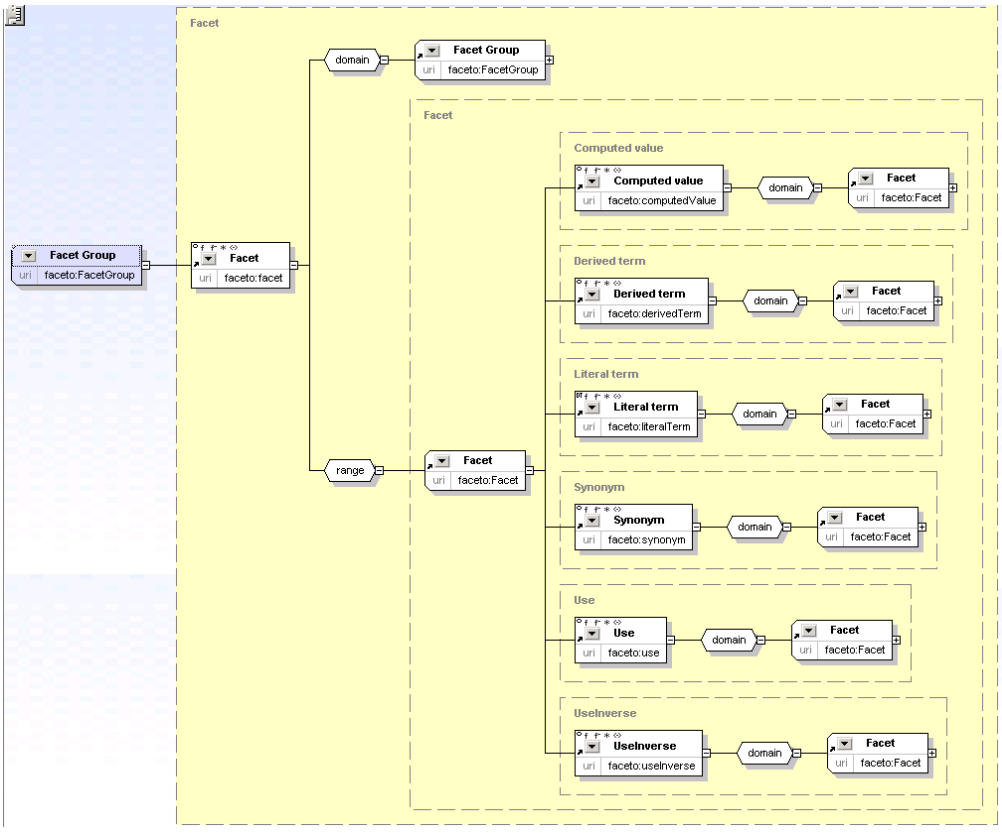


Figura 18 – Diagrama representando o vocabulário faceto.

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix faceto: <http://www.semanticnavigation.org/2008/faceto#faceto:> .

faceto:Facet rdf:type rdfs:Class ;
  rdfs:label "Facet" .
faceto:facet rdf:type owl:ObjectProperty .
faceto:FacetGroup rdf:type rdfs:Class ;
  rdfs:label "Facet Group" .
faceto:facet rdfs:domain faceto:FacetGroup ;
  rdfs:label "Facet" ;
  rdfs:range faceto:Facet .
faceto:use rdf:type owl:ObjectProperty ;
  rdfs:domain faceto:Facet ;
  rdfs:label "Use" .
faceto:useInverse rdf:type owl:ObjectProperty ;
  rdfs:domain faceto:Facet ;
  rdfs:label "UseInverse" .
faceto:literalTerm rdf:type owl:DatatypeProperty ;
  rdfs:domain faceto:Facet ;
  rdfs:label "Literal term" .
faceto:computedValue rdf:type owl:ObjectProperty ;
  rdfs:domain faceto:Facet ;
  rdfs:label "Computed value" .
faceto:expressionValue rdf:type owl:DatatypeProperty .
faceto:query rdf:type owl:DatatypeProperty ;
  rdfs:label "Query" .
faceto:derivedTerm rdf:type owl:ObjectProperty ;
  rdfs:domain faceto:Facet ;
  rdfs:label "Derived term" .
faceto:synonym rdf:type owl:ObjectProperty ;
  rdfs:domain faceto:Facet ;
  rdfs:label "Synonym" .
faceto:constraint rdf:type owl:DatatypeProperty .
faceto:word rdf:type owl:ObjectProperty ;
  rdfs:label "Word" .
faceto:literalValue rdf:type owl:DatatypeProperty ;
  rdfs:label "Literal Value" .
faceto:operation rdf:type owl:DatatypeProperty .
faceto:type rdf:type owl:ObjectProperty .
faceto:defaultWord rdf:type owl:ObjectProperty .
faceto:queryValues rdf:type owl:ObjectProperty
```

Quadro 23 – Vocabulário Faceto.

## 3.6. Geração Automática de Facetas

Aqui descreveremos um modelo de extração de facetas de uma base RDF. Nosso objetivo aqui não é propor um algoritmo ótimo mais simplesmente traduzir o modelo RDF para o modelo de navegação facetada.

A motivação para se desenvolver tal mecanismo de extração automática de facetas advém do fato de que o processo manual de identificação de facetas em uma base RDF é uma tarefa lenta e trabalhosa (Suominen et al., 2007). Além disso, determinar a relevância das facetas é inconcebível sem um algoritmo desta natureza. Também, tal algoritmo pode servir como uma ferramenta para geração automática das facetas na interface, sem a necessidade da intervenção manual de um projetista, permitindo assim que qualquer conjunto de triplas RDF possa ser facetado automaticamente.

Nos próximos tópicos, descreveremos detalhes de uma abordagem possível para geração automática das facetas baseada na teoria de informação, ganho de informação e entropia.

### 3.6.1. Definições Gerais

Como definido no início desse capítulo, uma faceta é uma tupla formada por termo/valores. O termo é sempre um predicado de uma tripla RDF ou a propriedade de um recurso. Por exemplo, local de nascimento, afiliação, idade, sexo, etc. Os valores da faceta são os objetos de uma tripla e são representados por uma lista de valores, ex., Rio de Janeiro, São Paulo, que também podem estar organizados em uma taxonomia. Por exemplo., País -> Estado > Cidade > Rio de Janeiro.

Em um sistema com um número elevado de facetas é necessário definir uma ordem de relevância entre as facetas para que a interface exiba primeiro o que for mais eficiente em termos de discriminação dos recursos sendo facetados.

Levando em consideração que uma faceta é equivalente a uma propriedade/objeto RDF, façamos a seguinte análise:

1. As propriedades que mais ocorrem são mais relevantes que as que menos ocorrem no conjunto total de triplas.
2. Caso uma propriedade contenha os mesmos valores em todos os recursos, essa propriedade não é muito discriminativa, dado que qualquer valor selecionado sempre retornará os mesmos recursos do conjunto inicial sendo facetado.

Utilizaremos a equação de entropia para determinar a relevância das propriedades. Basearemos no número de ocorrência de seus valores em cada recurso. Antes façamos as seguintes definições:

Sejam R os conjuntos de recursos de A

**Ocorrência (p) = |{x ∈ R / (x,p,y) ∈ A }|**

Quadro 24 – Definição de ocorrência

A fórmula acima determina quantas vezes a propriedade p ocorre distintamente no conjunto total de recurso. Note que só é contabilizada uma única ocorrência por recurso. O número máximo de Ocorrência(p) é igual ao número total de recursos sendo facetados.

Propriedades que ocorrem mais são mais relevantes do que as que ocorrem menos, dado que são capazes de filtrar um conjunto maior de recursos. Ou seja, se estamos facetando um conjunto com 100 recursos, mas uma determinada propriedade só ocorre em 40 deles, significa que a faceta baseada nesta propriedade só facetará no máximo 40 dos 100 recursos que desejamos filtrar.

A fórmula a seguir captura a propriedade acima e determina quantas vezes a propriedade p ocorre no conjunto total de recursos sendo facetados:

**OcorrênciaTotal(p)= |{(x,y,z) ∈ A / y = p}|**

Quadro 25 – Definição de ocorrência total.

Note que *OcorrênciaTotal(p)* pode ser maior que o número total de recursos sendo facetadas, dado que uma propriedade pode ocorrer duas vezes em um único recurso porém com valores distintos.

Outra informação importante que precisamos obter é referente à ocorrência dos objetos, ou valores das facetas. Um valor que ocorre em todos os recursos para uma data propriedade não deve ser utilizado como valor da faceta, dado que não é capaz de discriminar o conjunto facetado. A fórmula a seguir determina o número de vezes que um valor ocorre para uma propriedade em todos os recursos sendo facetados.

$$\textbf{OcorrênciaObj(p,o)= }|\{y \in R \,/\, (y,p,o) \in A\}|$$

Quadro 26 – Definição de ocorrência de um objeto.

O número máximo de *OcorrênciaObj(p,o)* tem que ser menor ou igual ao total de recursos sendo facetados.

$$\textbf{ProbObj(p,o) = OcorrenciaObj(p,o) / OcorrenciaTotal(p)}$$

Quadro 27 – Definição da probabilidade de ocorrência de um valor.

A probabilidade de ocorrências de propriedade *p* com o valor *o* e dado pelo número de triplas com a propriedade *p* igual a *o* dividido pelo número total de ocorrências de *p* em todos os recursos sendo facetados.

A equação da entropia de uma propriedade p é dada por:

$$\textbf{H(p)= } -\sum_{o \in R}^{n} \Pr obObj(p,o) \log(\Pr obObj(p,o))$$

Quadro 28 – Fórmula da entropia.

Abaixo segue um conjunto de propriedades que podem ser observadas da equação da entropia acima e que serão utilizadas para determinarem a relevância de uma faceta.

Quando a entropia H(p) = 0 significa que todas as instâncias sendo facetadas possuem os mesmos valores de *o* quando a propriedade *p* ocorre.

Isso implica que essa propriedade é pouco discriminativa, pois qualquer valor selecionado nesta faceta levará ao mesmo conjunto inicial sendo facetado. Portanto, concluímos que propriedades com entropia 0 não devem ser utilizadas como faceta.

Se um valor *o* ocorre para uma propriedade em todos os recursos, então este valor deve ser descartado dos valores possíveis para a faceta. O valor *o* só deve ser eliminado dos valores possíveis da faceta após o cálculo da entropia.

A entropia H(p) é máxima quando existe uma distribuição equivalente entre os valores de uma propriedade para um determinado conjunto de recursos.

A entropia máxima garante o conjunto mais discriminativo. Os conjuntos com a entropia máxima devem aparecer no topo da lista das facetas.

### 3.6.2. Algoritmo

Os passos para se extrair as facetas são dados pelo algoritmo a seguir.

1. Determine a Ocorrência(p)
2. Determine a OcorrênciaTotal (p)
3. Determine a OcorrênciaObj (p,o)
4. Determine H(p)
5. Ordene todos p por Ocorrência(p);
6. Ordene todos p que possuam a mesma Ocorrência(p) pela entropia máxima;
7. Remova p que possua H(p) = 0;
8. Para cada *p* determine *o*;
9. Remova todo *o* tal que OcorrênciaObj(*p*,*o*) = Ocorrencia(p);
10. Remova todo *p* tal que OcorrênciaObj(*p*,*o*) = 1, para todo *o*.

Tal algoritmo dará todas as propriedades e objetos ordenados pela sua relevância segundo os critérios discutidos acima. Com essas informações em mãos podemos criar o termo e valores das facetas e exibi-las na interface em ordem de relevância.

## 3.7. Conclusão

Neste capítulo propomos um modelo de navegação facetada baseado no modelo RDF. Para tal modelo definimos uma linguagem de especificação de facetas e um algoritmo para extração automática de facetas de uma base RDF.

A linguagem de especificação de facetas permite ao projetista expressar as facetas em função do seu próprio domínio RDF. Ela separa detalhes da construção das facetas, sendo totalmente independente do modelo de operação aplicado à navegação facetada (interseção, união, etc.). Tal linguagem permite a definição de facetas computadas, baseada na combinação de valores e propriedades do modelo RDF, algo que não foi contemplado em nenhum modelo até então elaborado.

O algoritmo de extração automática de facetas nos permite facilmente construir uma aplicação facetada sobre uma base RDF. Tal mecanismo é capaz de extrair as facetas de uma base RDF que é o passo mais complexo na construção de uma aplicação facetada. Obviamente, outras abordagens de extração automática de facetas são possíveis, como a descrita por (Oren, 2006). O objetivo deste trabalho não foi desenvolver o algoritmo ótimo, mas sim criar uma ferramenta que pudesse ser facilmente incorporada no Explorator.

Uma futura evolução deste trabalho seria levar em consideração as relações definidas na ontologia dos dados RDF para a identificação de uma taxonomia dos valores das facetas. Organizar os valores das facetas em uma taxonomia é útil quando uma faceta possui um número elevado de valores para serem exibidos na interface. Porém, técnicas como paginação e rolagem de página também podem ser aplicadas para se resolver o mesmo problema.