4
A Ferramenta Explorator

Estudos empíricos demonstram que os usuários têm um melhor desempenho e maior satisfação quando podem visualizar e controlar a sua pesquisa (Rossi et al., 1998). Este é um dos principais objetivos da Explorator: colocar os usuários no controle de suas consultas, e fornecer uma resposta imediata às suas ações. Para tanto, basearemos nossa interface no paradigma de manipulação direta (Shneiderman, 1983) e query-by-example (Zloof, 1977).

4.1.
Manipulação direta

O paradigma de manipulação direta (Shneiderman, 1983) é um tipo de interação homem-computador que envolve a representação de objetos, e ações incrementais sobre eles, que forneçam uma resposta imediata. Um exemplo de manipulação direta é o redimensionamento de um retângulo, puxando seus cantos ou arestas com um mouse.

O sucesso da manipulação direta advém da constante visibilidade dos objetos de interesse na interface, e do fato de cada ação sobre eles produzirem um resultado compreensível dentro do domínio da tarefa que é imediatamente visível na interface. A aproximação entre o domínio da tarefa e o domínio da interface diminui o stress e carga mental do operador da tarefa.

De acordo com Shneiderman, a manipulação direta funciona melhor do que linha de comando porque ajuda o usuário a resolver o problema diretamente. No entanto, uma representação adequada é fundamental para a resolução de problemas e para a aprendizagem do usuário.

Tal paradigma fornece os elementos necessários para criarmos uma interface que suporte nosso modelo de operações e o modelo RDF, exigindo o mínimo de conhecimento do usuário para operá-lo.

Quando falamos em criar um mecanismo genérico de exploração RDF, basicamente estamos tratando de um sistema que exibe e manipule triplas e recursos. No entanto, permitir o usuário somente visualizar tais triplas e recursos e navegar sequencialmente entre elas, não garante um mecanismo de exploração eficiente. Tal abordagem já foi utilizada nas ferramentas de navegação RDF existentes. Acreditamos que a chave para um mecanismo de exploração adequado é permitir ao usuário manipular tais elementos na interface. Assim, herdaremos os benefícios que citamos da manipulação direta.

4.2. Query-by-example

Query-by-example (Zloof, 1977) é um paradigma de consulta. O cerne desse paradigma é criar um mecanismo de consulta através da analogia com um exemplo especificado pelo usuário. Este exemplo tipicamente reflete o modelo mental do usuário para a tarefa. Para o modelo de dados relacional, Zloof (Zloof, 1977) utilizou uma tabela bidimensional, onde uma consulta é formulada preenchendo os espaços em branco da tabela com exemplos de solução. O resultado da consulta é baseado nos registros que correspondem ao padrão da tabela preenchida pelo usuário.

Dado que o Explorator é destinado para usuários que tenham tido algum contato com o modelo RDF, acreditamos que o modelo mental do usuário, quando esta lidando com bases RDF, é o modelo triplas sujeito-predicado-objeto. No entanto, o modelo de grafo também pode ser considerado, dado que no fundo um conjunto de triplas relacionam sujeitos e objetos, criando um grafo dirigido de informação. Um dos nossos objetivos é tentar prover um mecanismo de consulta que utilize as operações definidas em nosso modelo de operações, dispostas em uma interface visual o mais próximo possível de seu modelo mental.

Nossa solução de QBE partirá do modelo mental do usuário associado ao modelo de triplas. Basicamente, nossa solução permitirá ao usuário criar expressões SPO visualmente e incrementalmente. Tais expressões serão posteriormente traduzidas em uma linguagem de consulta RDF. Análogo ao que foi feito por Zloof, estaremos dando aos usuários um modelo visual, onde ele será capaz de reconhecer seu modelo mental e prover exemplos da realidade que deseja obter. E a cada passo, ele estará produzindo expressões descritas no nosso

modelo de operações. A característica incremental de nossa solução advém mais do paradigma de manipulação direta, em que a cada passo dado na construção do grafo, ou na composição da tripla, ele poderá visualizar um resultado intermediário de sua consulta.

A interface do Explorator engloba dois elementos bem definidos. Um é a representação dos recursos, triplas e conjuntos, na interface. O outro é o mecanismo de consulta que disponibilizamos aos usuários. Para o primeiro caso, aplicamos os conceitos envolvidos no paradigma da manipulação direta. No segundo caso, aplicaremos os conceitos envolvidos no paradigma de QBE.

4.3. Recursos, triplas e conjuntos

Nosso modelo de operação manipula 3 tipos de elementos que são suportados pela interface do Explorator: recursos, triplas e conjuntos.

Na construção dos *widgets* que representam tais elementos mantivemos em mente os seguintes princípios do paradigma de manipulação direta:

- A natureza da relação entre a linguagem de entrada e saída na interface deve ser tal que a saída de uma expressão pode servir como um componente de entrada para outra expressão.
- O sistema deverá fornecer representações de objetos que se comportem como se fossem os próprios objetos de interesse.

Levando esses princípios em consideração e a natureza da aplicação em questão, criamos as três representações abaixo para um recurso, uma tripla e um conjunto, respectivamente:

Todo recurso é representado na interface pelo seu *rdfs:label* ou pela parte local da URI do recurso (*localname*).

Figura 19 - Representa o recurso Product na interface.

Uma tripla é representada na interface pelos 3 recursos que a formam. A propriedade e o objeto das triplas foram aninhamos como uma forma de destacar a qual sujeito elas pertencem.

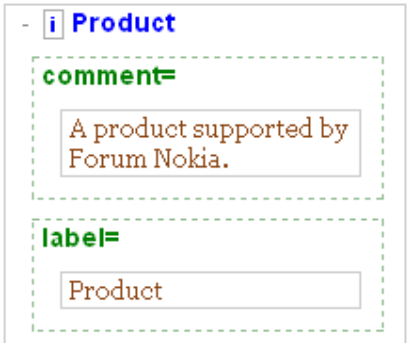

Figura 20 – Representação das duas triplas (Product, comment, 'A product supported by Forum Nokia') e (Product, label, 'Product') , na interface.

Um conjunto é formado por um conjunto de triplas. No entanto, num dado momento, o usuário é capaz de visualizar ou os sujeitos, ou os predicados, ou os objetos das triplas, ou as triplas em si.

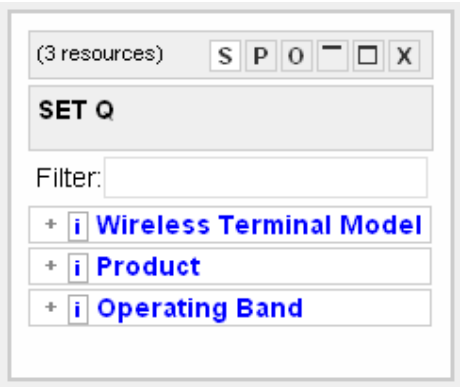

Figura 21 - Representa o conjunto dos recursos Wireless Terminal Model, Product e Operating Band na interface.

Com essas representações de recursos, triplas e conjuntos, damos ao usuário o acesso direto ao objeto de interesse, que no nosso caso, são as informações descritas em RDF. Veremos mais à frente quando falarmos dos *widgets* das operações do modelo, que o usuário obtém como resultado das expressões que compõe, um conjunto de *widgets* na mesma linguagem que

utilizou como entrada nas expressões. Dessa forma, garantimos ao usuário que ele manipulará na interface diretamente os elementos que o sistema representa.

4.4. Classes, propriedades, literais e recursos

O modelo RDF não se resume a recursos e triplas. Classes, propriedades e valores literais são outros primitivos definidos nesse modelo. É fundamental que a interface distinga de forma gráfica tais primitivos, facilitando a tarefa do usuário de identificá-los.

Iremos utilizar cores para distinguir tais tipos de recursos na interface. Na figura abaixo, temos um exemplo da distinção entre um recurso, um predicado, uma classe e um literal, que são destacados em preto, verde, azul e marrom, respectivamente.

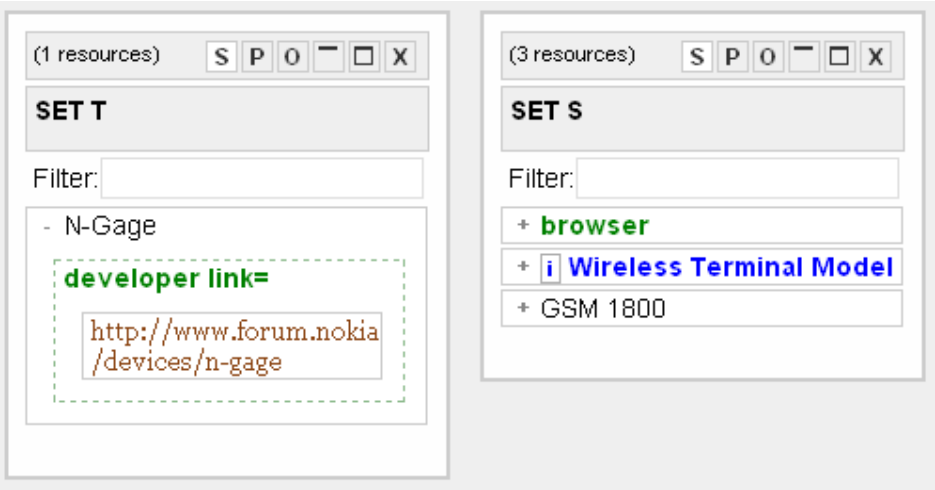

Figura 22 - tipos de recursos na interface. A cor azul indica recursos do tipo Class, verde do tipo Property e marrom do tipo literal.

4.5. Modelo de seleção

Descreveremos aqui uma parte importante da interface que é o modelo de seleção. O modelo de seleção definirá quais e como os recursos, triplas e conjuntos poderão ser selecionados para serem utilizados na construção de expressões dentro do domínio do modelo de operações.

Basicamente, o modelo de operações opera sobre recursos. Apesar de estarmos visualizando um conjunto de triplas na interface, ou projeções no sujeito,

predicado ou objeto destas triplas, todas as operações do modelo são sobre um conjunto de recursos. Dessa forma, nosso modelo de seleção na interface permite selecionarmos um recurso ou um conjunto deles de forma arbitraria.

Abaixo descreveremos as possibilidades do modelo de seleção. Um objeto selecionado na interface é marcado com bordas tracejadas em azul, como mostra a seleção de um conjunto abaixo.

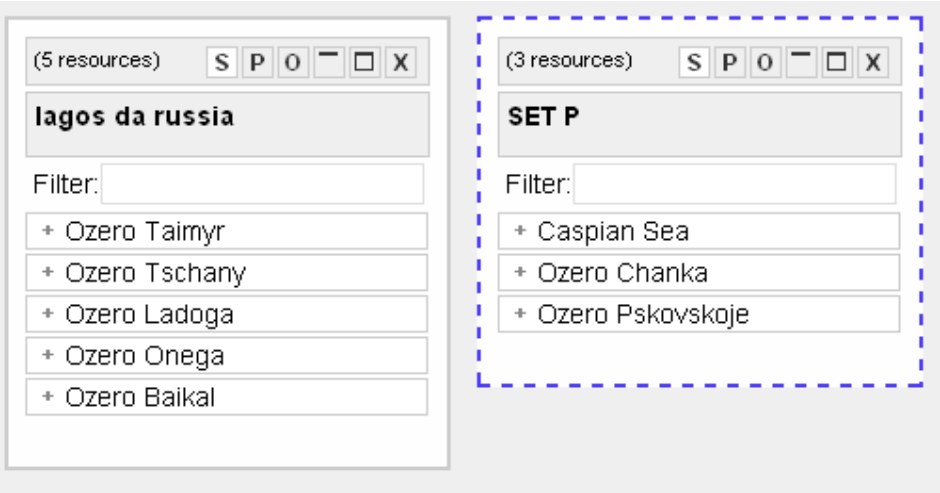

Figura 23 - Seleção de um conjunto que possui três recursos (Caspian Sea, Ozero Chanka,Ozero Pskovskoje).

Ao selecionar um conjunto na interface, o usuário forma exatamente o conjunto dos recursos que esta visualizando na tela. Abaixo temos a seleção de um recurso.

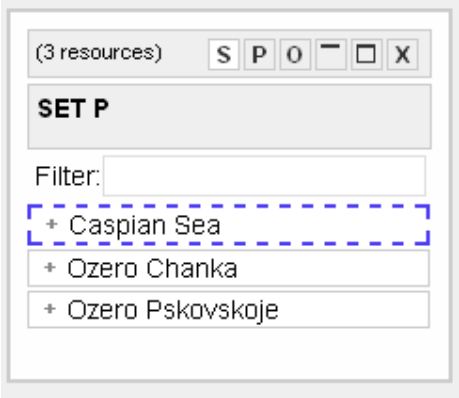

Figura 24 - Seleção de um único recurso (Caspian Sea).

Na Fig. 25 A, temos uma representação da seleção de predicados de uma tripla. Note que ambas as seleções indicam a seleção somente do predicado da tripla. Já a Fig. 25 B representa a seleção de objetos. Podemos tanto selecionar o conjunto de objetos de uma tripla assim como um objeto individualmente. Note, no entanto, que na Fig. 25 A e B temos a seleção de um conjunto de recursos (predicados e objetos) das triplas.

A B

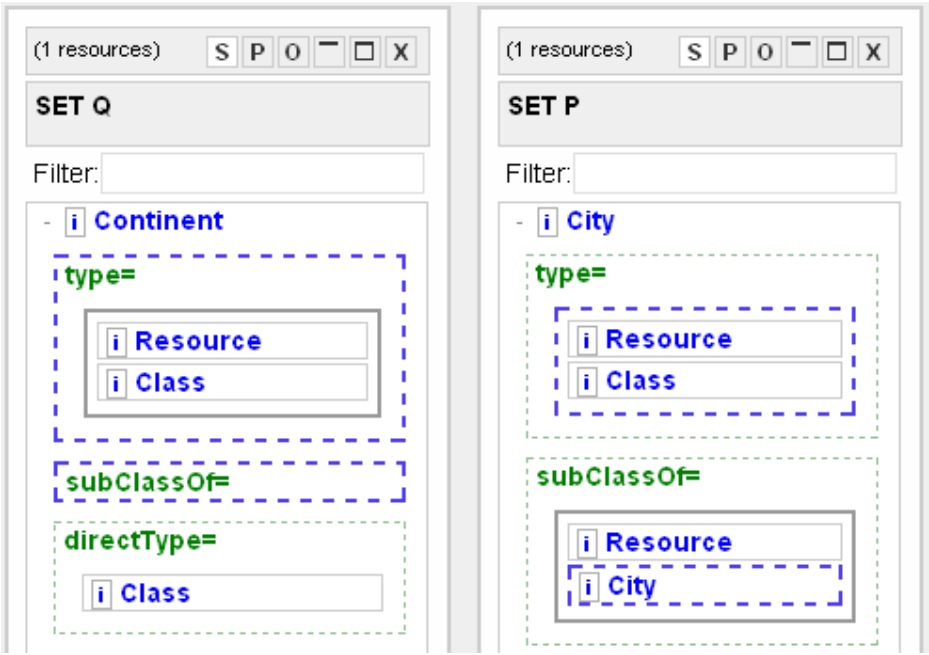

Figura 25 – Modelo de seleção de predicado e objetos. A fig. A representa a seleção de propriedades das triplas, e a fig. B a seleção de objetos das triplas.

O modelo de seleção descrito acima garante ao usuário poder manipular diretamente o item de informação que está visualizando.

4.6. Operações do modelo na interface

Como discutimos em tópicos anteriores, a melhor forma de suportarmos nosso modelo de operações na interface é utilizarmos o paradigma de *Query-by-example*. São dois grupos de operações que desejamos suportar na interface: operações sobre conjunto e operações de consulta.

4.6.1.
Operações sobre conjunto

Para as operações sobre conjunto acreditamos que a melhor metáfora a ser implementada na interface é a de uma calculadora com operadores infixados. As operações de união, interseção e diferença de conjunto exigem que sejam definidos dois conjuntos que serão utilizados na operação, que são identificados através de seleções, segundo o modelo já descrito. Basicamente o usuário precisará realizar a seguinte seqüência de passos:

1. Selecionar um conjunto de recursos
2. Selecionar um operador (união, interseção ou diferença)
3. Selecionar outro conjunto de recursos
4. Obter o resultado.

Acreditamos que esse modelo de interação na interface é o mais intuitivo para o usuário e o que mais aproxima seu modelo mental da tarefa que irá realizar na interface. Dessa forma, criamos os seguintes *widget,* na interface do *Explorator*:

Figura 26 - Operadores de união, interseção e diferença.

Observe-se que o operador “=” segue a metáfora de calculadora; ao ser ativado, efetua a operação previamente selecionada.

A mesma metáfora de calculadora pode ser utilizada para a operação SPO do nosso modelo. No entanto, vale ressaltar que essa metáfora é aplicável quando consideramos o modelo mental do usuário como um modelo de triplas. Uma metáfora distinta será utilizada para o caso em que o modelo mental está associado a um grafo.

4.6.2.
Operações de consulta SPO

Assim como operações sobre conjunto no nosso modelo, a operação de consulta SPO também pode ser vista como uma seqüência de passos que o usuário deve reproduzir na interface, envolvendo seleção de conjunto de recursos e

operadores. Basicamente, para criar uma consulta SPO o usuário precisa definir no mínimo um conjunto de recursos e um parâmetro (S,P ou O). Utilizaremos a mesma metáfora da calculadora para representarmos essa operação no nosso modelo na interface. O *widget* abaixo foi criado para representar a operação SPO na metáfora da calculadora.

Figura 27 - Operação SPO na interface do Explorator.

Os mecanismos discutidos acima estão dispostos na barra de ferramenta do *Explorator*, como mostrado abaixo.

Figura 28 - Barra de ferramenta do Explorator.

Para exemplificar a operação SPO, suponha que tenhamos os recursos *LocateIn* e *Kazakstan* na interface do Explorator:

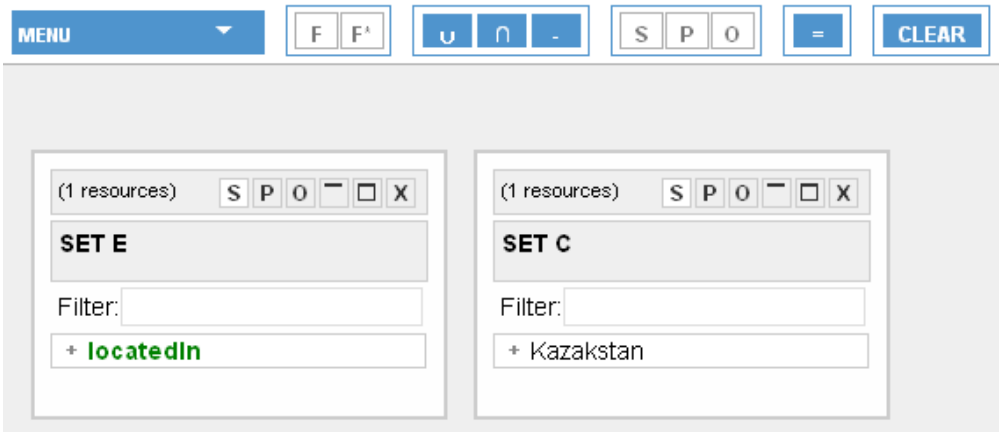

Figura 29 – Recursos locatedin e kazakstan.

Nesse momento, o usuário decide obter tudo que esteja localizado no país Kazaquistão. O primeiro passo para elaborar tal consulta é clicar no recurso *locatedIn* para selecioná-lo.

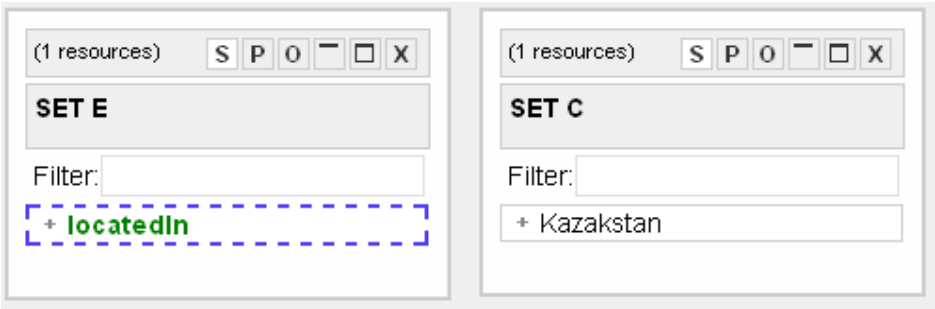

Figura 30 – Seleção do recurso locatedIn.

Logo em seguida ele clica no operando [P] na barra de ferramentas. Neste instante, o recurso *locatedIn* e o operando [P] são destacados em cinza, como mostra a figura abaixo:

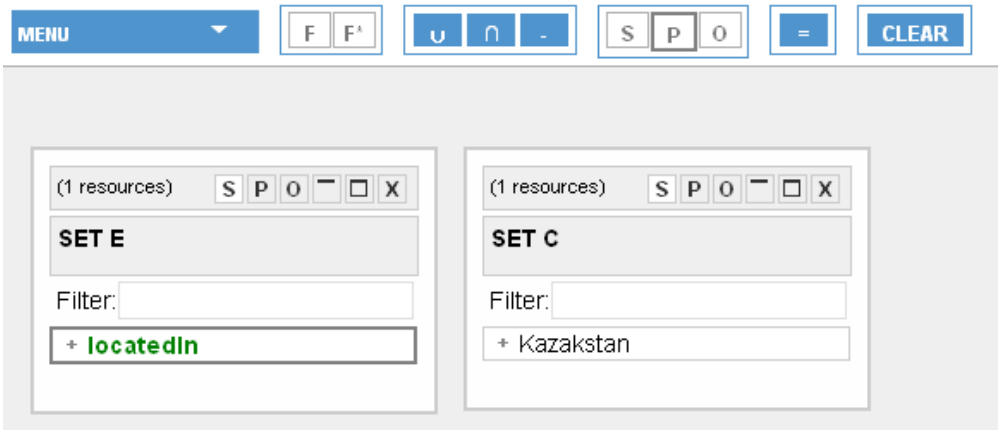

Figura 31 – Atribuição do recurso locatedin ao parâmetro P da operação SPO.

Em seguida, como feito com o recurso *locatedIn*, o usuário clica no recurso *Kazakstan* e logo depois clica no operando [O] na barra de ferramentas. Nesse instante, o recurso *Kazakstan* e o operando [O] são destacados de vermelho.

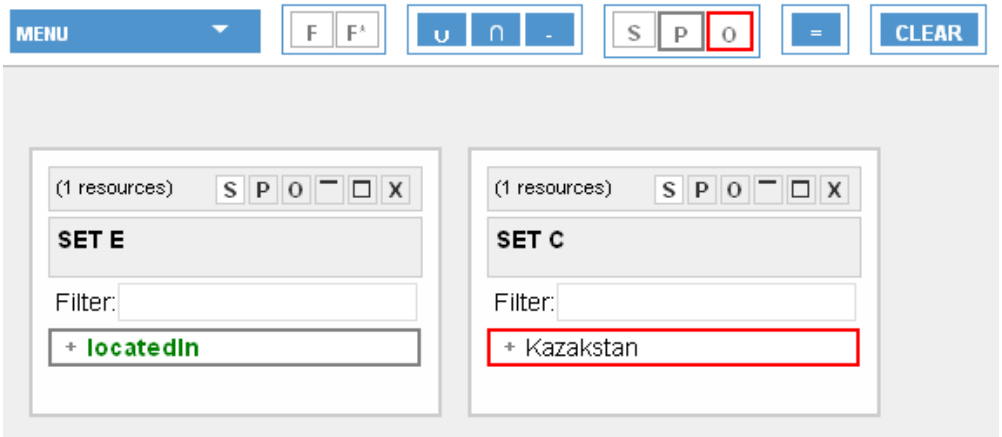

Figura 32 - Atribuição do recurso kazakstan ao parâmetro O da operação SPO.

O último passo do usuário é clicar no operador [=], para obter o resultado de sua consulta, mostrado na figura abaixo:

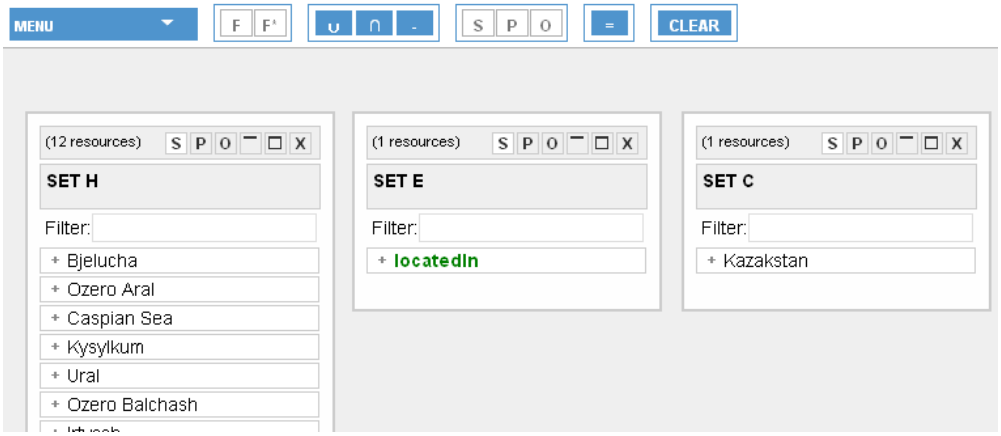

Figura 33 – Resultado da operação SPO

4.7. Repositórios

O Explorator permite navegarmos em qualquer *SPARQL Endpoint* existente na Web. Existem alguns já pré-configurados no *Explorator* que podem ser utilizados. No entanto, caso o usuário queira adicionar outro, também é possível.

No menu da barra de ferramentas existe a opção *Repository*, onde o usuário pode acessar a interface de configuração de repositórios. Um repositório no *Explorator* representa um *SPARQL Endpoint*.

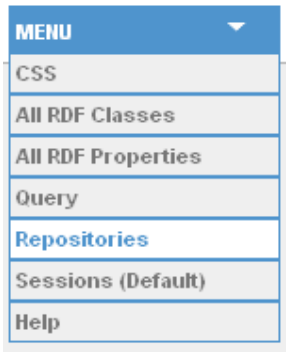

Figura 34 – Menu de acesso a interface de configuração de repositórios.

Na interface de configuração de repositórios, o usuário pode habilitar ou desabilitar um repositório existente, ou adicionar um novo. Todo repositório habilitado são pesquisados nas consultas dos usuários. Ou seja, se tivermos dois repositórios habilitados, por exemplo Nokia e Mondial, caso o usuário clique na operação do menu *All RDF Classe*, o *Explorator* irá retornar a união das classes existentes tanto no repositório Nokia, quanto no repositório Mondial.

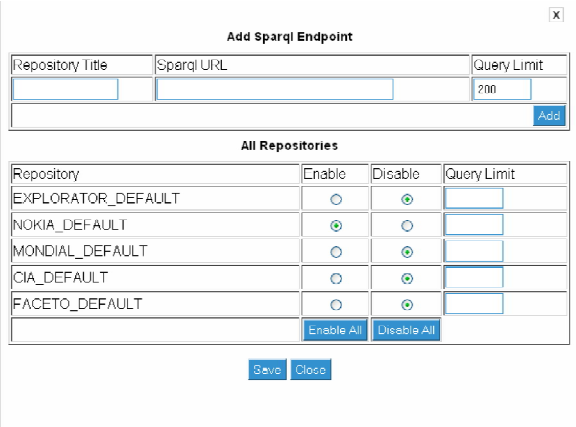

Figura 35 – Interface de configuração de repositórios do Explorator.

Para adicionar um novo repositório, o usuário precisa definir um título para o repositório, fornecer a URI do *SPARQL Endpoint*, e definir um limite para a consulta. O limite para consulta é um valor que será utilizado na cláusula SPARQL *limit*. Esse limite pode ser necessário para garantir um bom desempenho durante a exploração. Note, entretanto, que ele restringirá o número de triplas que serão retornados. O limite pode ser alterado em qualquer repositório existente.

Na figura a seguir definimos um repositório para o DBpedia[44] com um limite de 200.

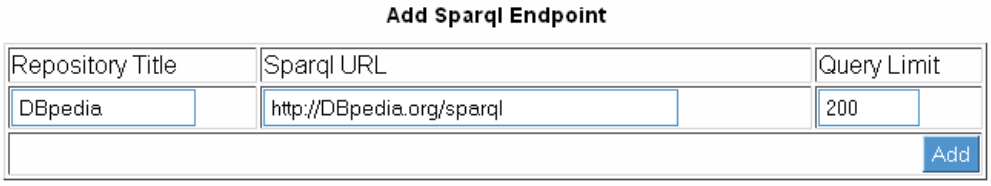

Figura 36 – Exemplo da adição do Sparql Endpoint.do DBpedia.

4.8. Navegação de-referenciada

O *Explorator* também permite a navegação através de de-referenciação de URIs. Como discutimos anteriormente, existem *browsers* (ex. Disco) que só

[44] http://wiki.dbpedia.org/OnlineAccess

navegam na Web Semântica através deste mecanismo. Para utilizar esse mecanismo no *Explorator* basta o usuário fornecer uma URI e clicar no botão "GO" na barra de endereço do *Explorator*.

Figura 37 – Barra do Explorator evidenciando a barra de endereço.

Ao clicar no botão 'GO', o *Explorator* faz o download do arquivo RDF de-referenciado e carrega as triplas contidas neste arquivo em um repositório destinado a este fim (denominado: EXPLORATOR_DEFAULT). A partir deste momento, o usuário pode explorar as informações contidas neste repositório, como se fosse qualquer outro repositório, ou seja, utilizando todas as operações existentes.

4.9. Navegação Facetada

Descrevemos aqui a interface de navegação facetada implementada pelo *Explorator*. Como vimos no capítulo 3, uma interface facetada possui três estruturas básicas de informação: **o conjunto sendo facetado**, as **facetas** e as **facetas selecionadas**. Na interface do *Explorator*, o conjunto sendo facetado é um conjunto qualquer na interface que tenha sido facetado através da operação F ou F* existente na barra de ferramentas.

Figura 38 – Destaque das operações F e F* para facetar um conjunto de recursos.

A operação F aplica um conjunto de facetas pré-definidas pelo usuário, descritas no vocabulário FACETO. A definição destas facetas deve estar previamente disponível em algum dos repositórios habilitados do *Explorator*. Já a operação F* determina automaticamente todas as possíveis facetas para o

conjunto selecionado a ser facetado, seguindo a heurística de entropia apresentada anteriormente.

Para executar essas operações é necessário selecionar um único conjunto na interface e clicar na operação F ou F* na barra de ferramentas. A figura abaixo mostra a interface de navegação facetada que será exibida ao usuário após a execução de uma dessas operações.

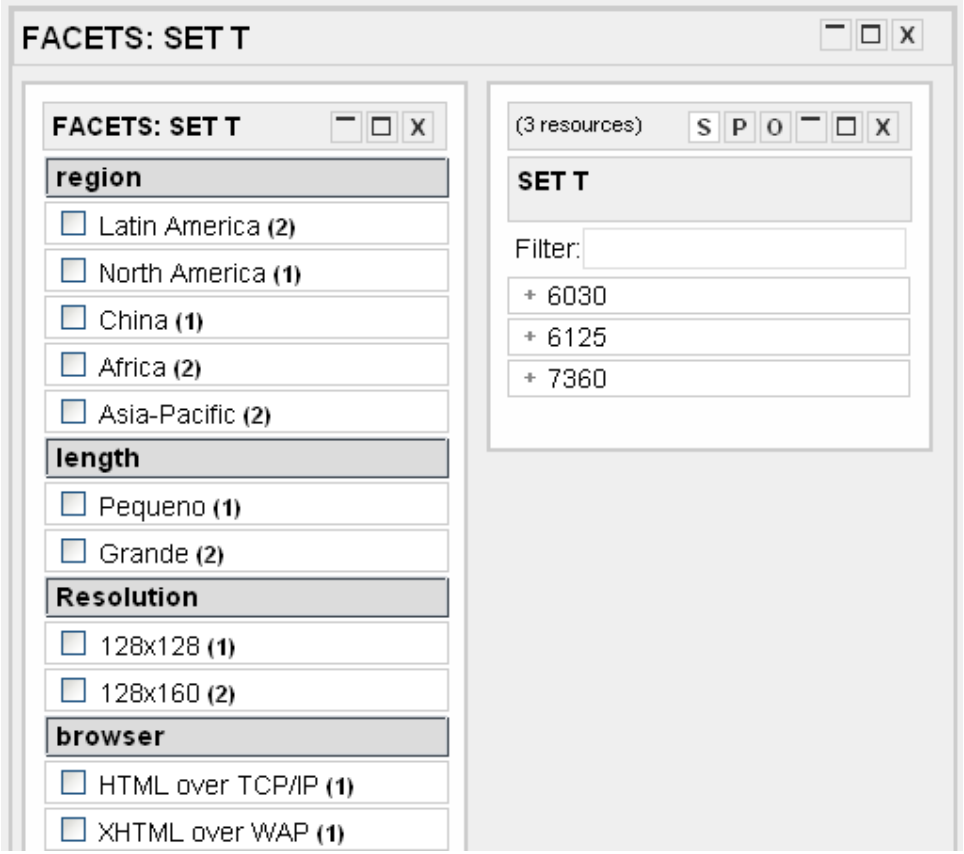

Figura 39 – Interface de navegação facetada do Explorator. O conjunto da esquerda representa as facetas e o conjunto da direita os elementos sendo facetados.

Note que os *checkboxes* nesta interface representam tanto **as facetas** quanto as **facetas selecionadas**, dado que uma vez clicado em qualquer um deles estaremos aplicando a faceta ao conjunto e indicando o valor que foi selecionado. O conjunto da esquerda representa as facetas e o conjunto da direita os elementos sendo facetados. O usuário pode utilizar o *checkbox* na esquerda para selecionar os valores de uma faceta. Note, que podem ser selecionados mais de um valor do

checkbox, o que para uma mesma faceta representa a união de valores, ou seja, se for selecionado os valores IM e MMS para a faceta *Messaging*, o conjunto resultante será o conjunto composto pelos recursos que possuam a propriedade *Messaging = IM* ou *Messaging = MMS*.

4.10. Outras operações na interface.

Outra operação que o usuário possui na interface, além das discutidas até o momento, é a operação relacionada à visualização dos recursos de uma tripla. Tal operação permite ao usuário visualizar os sujeitos, predicados ou objetos de uma tripla isoladamente. Podemos acessá-la no cabeçalho da caixa de cada conjunto, como mostra a figura abaixo.

Figura 40 - Cabeçalho da caixa de conjunto

No exemplo mostrado na figura abaixo, estamos visualizando os sujeitos das triplas no conjunto Z, os predicados do conjunto Y e os objetos do conjunto X.

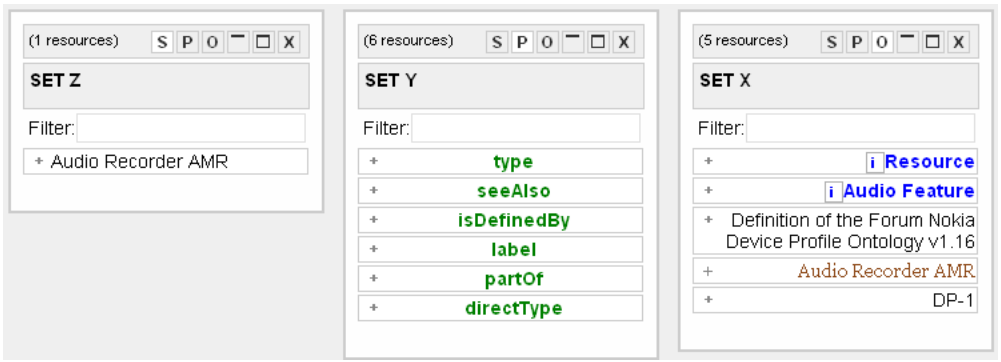

Figura 41 - Exemplos das diferentes visualizações de uma tripla.

Outras operações disponíveis na interface são os menus de atalho. Os menus de atalho são consultas predefinidas que foram evidenciadas no experimento que realizamos com os usuários como sendo de uso constante.

Dependendo do tipo do recurso sendo exibido, diferentes opções são exibidas. Ou seja, para os recursos do tipo *rdfs:property* temos a consulta *UsedBy* – que retorna todos os recursos que usam tais propriedades – e a operação *HasValues* – que retorna todos os possíveis valores dessa propriedade. Para os recursos do tipo *rdfs:class*, temos a consulta *Instances* – que retorna todas as instâncias da classe. Já para recursos em geral, temos a operação *AllProperties* – que exibe todas as propriedades do recurso – e a operação *ObjectOf* que retorna todos os recursos que este recurso aparece como objeto na tripla.

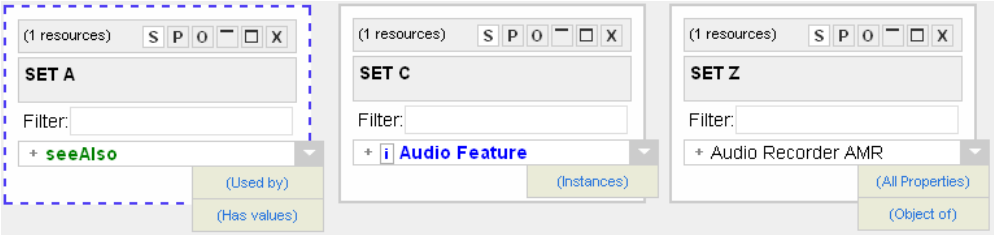

Figura 42 – Menu de atalhos de um recurso.

4.11. Resumo das operações do Explorator

Para capacitar os usuários em sua tarefa de exploração, o *Explorator* suporta as seguintes operações na interface:

- Buscar todos os recursos contendo um determinado texto (usando a caixa de busca na barra de ferramentas);
- Selecionar recursos, ao clicá-los;
- Obter detalhes de um recurso, dando duplo clique para revelar todas suas as triplas em que o recurso é sujeito;
- Selecionar múltiplos recursos e conjuntos, executando ctrl+clique;
- Obter um novo conjunto pela união, interseção ou diferença entre outros dois, através da função correspondente na barra de ferramenta;
- Obter um novo conjunto através de uma expressão SPO;

- Mudar a visualização de um conjunto de recursos, isto é, agrupando as triplas pelo sujeito, predicado ou objeto. Essa operação é realizada utilizando a função corresponde no cabeçalho de cada conjunto;
- Adicionar e remover novos Sparql Endpoints para a exploração dos usuários;
- Renomear um conjunto na interface.

4.12. Cenário de Exemplo

Vamos agora ilustrar o uso do *Explorator*. Suponha que David, um geógrafo, precise encontrar todos os lagos contidos inteiramente dentro do território Russo (e em nenhum outro país). Tal tarefa pode ser decomposta nos seguintes passos:

1. Encontre todos os lagos na base de dados;
2. Encontre o país Rússia;
3. Encontre todos os lagos da Rússia, obtendo um conjunto que chamaremos de LR;
4. Encontre os países que fazem fronteira com a Rússia;
5. Encontre os lagos que estão localizados nos países vizinhos da Rússia, obtendo um conjunto que chamaremos de LN, e;
6. Construa o conjunto dos lagos contidos exclusivamente na Rússia, calculando a diferença entre LR e LN (LR - LN).

Para encontrar todos os lagos na base RDF, David busca por ‘lago’:

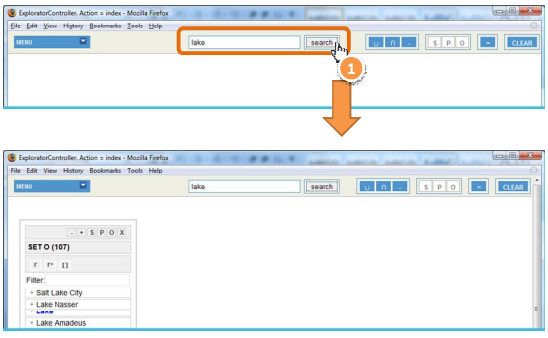

Figura 43 – Pesquisa pelos lagos.

Ele localiza a classe Lake (lago) no conjunto resultante, e obtém o conjunto de instâncias da classe Lake clicando no link Instances (instâncias), que são todos os lagos contidos na base.

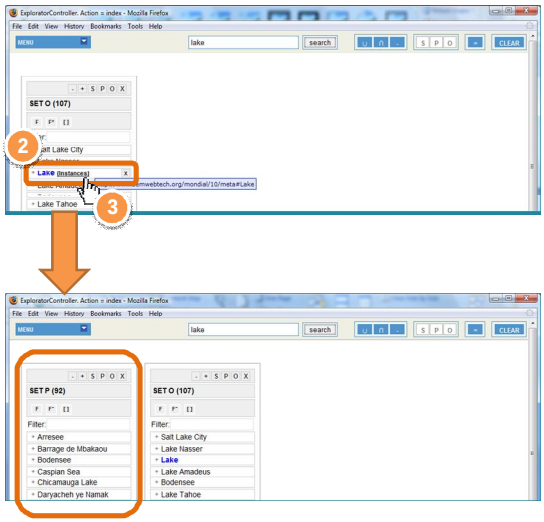

Figura 44 – Instancias de lagos

No próximo passo, para encontrar Rússia, ele busca por “russia” e localiza o recurso Rússia no conjunto resultante:

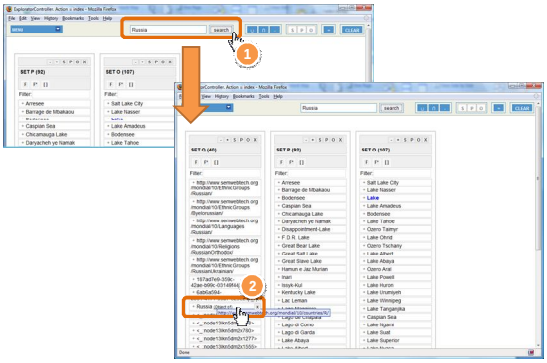

Figura 45 – Pesquisa por Russia

Para garantir que ele obteve o recurso certo, ele visualiza os detalhes do recurso:

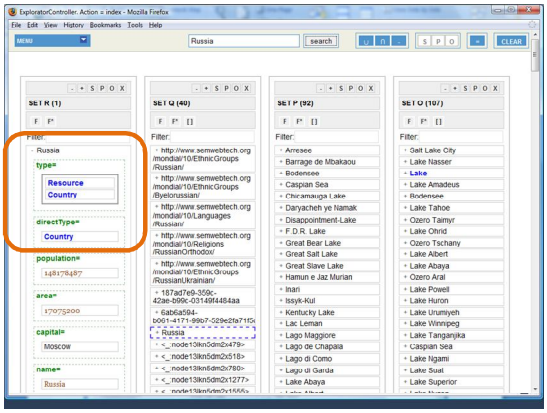

Figura 46 – Detalhes do recurso Rússia

No próximo passo, para encontrar os lagos LR da Rússia, ele seleciona o conjunto de lagos (1) e marca como sujeito da consulta clicando no botão [S] na barra de ferramentas (2).

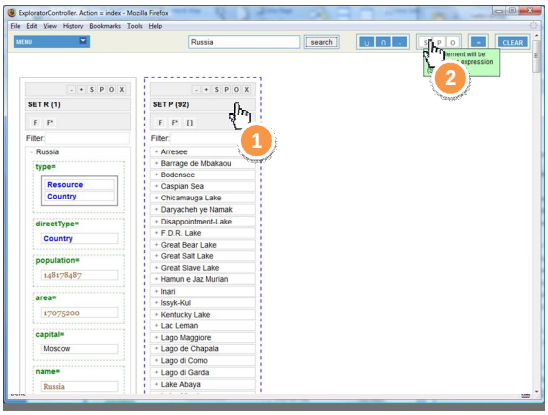

Figura 47 – Exemplo de uso da operação SPO

Continuando a contruir a consulta, ele seleciona o recurso Rússia (3) e marca-o como objeto da consulta clicando no botão [O] na barra de ferramentas (4):

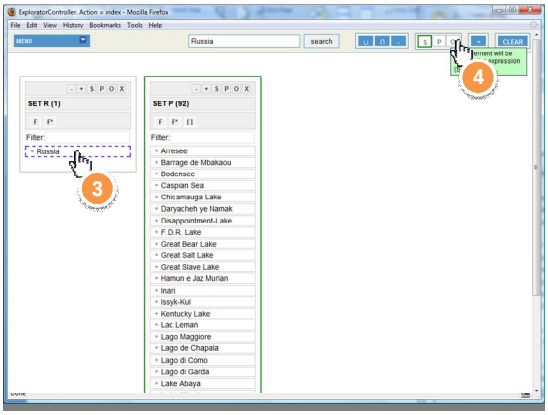

Figura 48 – Exemplo de uso da operação SPO

Ele executa a consulta para obter o conjunto de todos os lagos da Rússia clicando no botão [=] na barra de ferramentas (5):

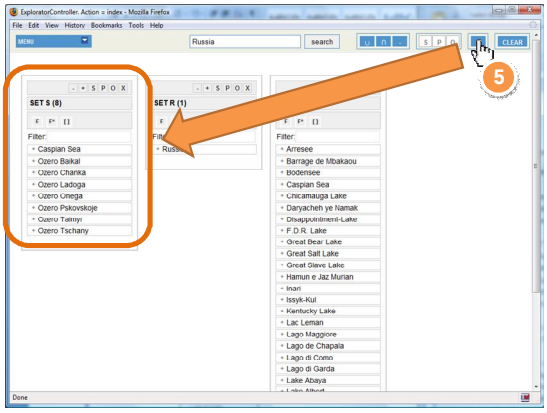

Figura 49 – Lista de lagos da Russia.

No próximo passo, para encontrar os paises que fazem fronteira com a Rússia, ele visualiza os detalhes do recurso Rússia e localiza a propriedade vizinho (neighbor), dessa forma encontrando os seus vizinhos:

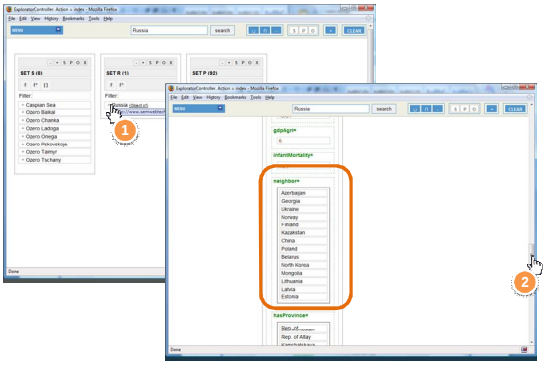

Figura 50 – Lista de vizinhos da Rússia.

Para encontrar todos os lagos vizinhos da Rússia, ele seleciona o conjunto de lagos da Rússia e marca-o como sujeito da consulta, como feito nos passos anteriores:

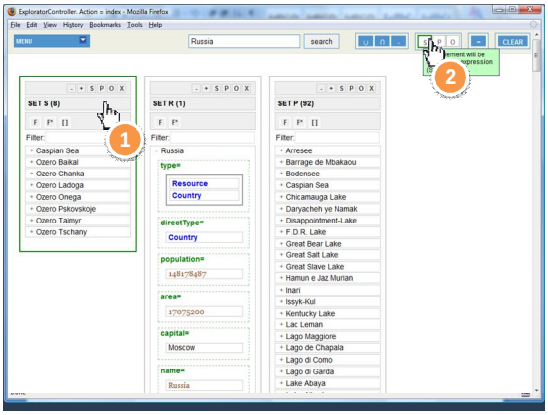

Figura 51 – Obtendo os lagos dos vizinhos.

Ele seleciona o conjunto de vizinhos da Rússia e marca-o como o objeto da consulta:

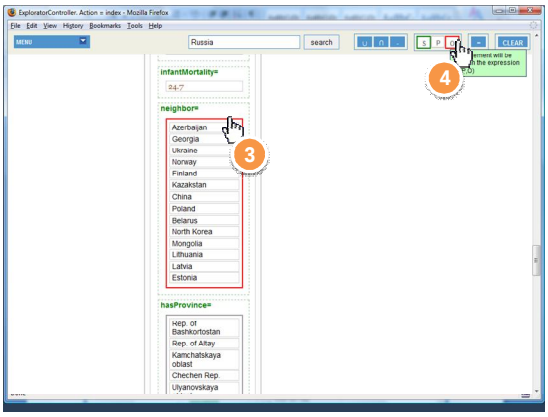

Figura 52 – Obtendo o lagos dos vizinhos da Rússia com a operação SPO.

Ele então executa a consulta para encontrar todos os lagos vizinhos da Rússia:

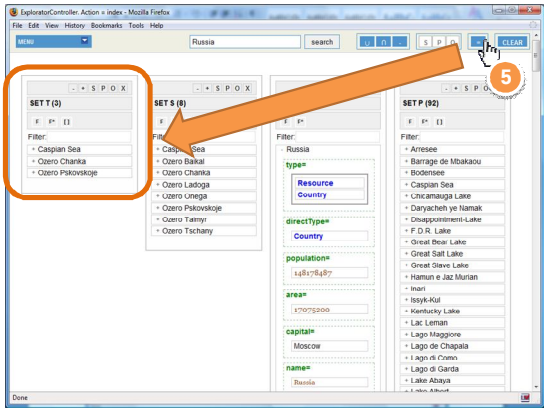

Figura 53 – Lista de lagos contidos nos países vizinhos da Rússia e na Rússia.

Finalmente, para construir um conjunto de lagos contidos exclusivamente na Rússia, ele precisa calcular a diferença entre o conjunto de lagos da Rússia e o conjunto de lagos pertecentes aos países vizinhos da Rússia. Para fazer isso, ele seleciona o primeiro conjunto (1) e marca o operador de diferença na barra de ferramentas(2):

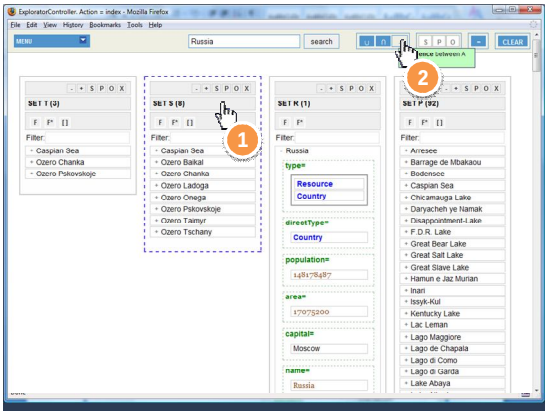

Figura 54 – Obtendo a interseção entre os lagos da Rússia e dos vizinhos.

Por fim, ele seleciona o segundo conjunto (3) (contendo os lagos dos vizinhos da Russia) e executa a diferença clicando no botão '=' barra de ferramentas (4), dessa forma obtendo o conjunto desejado:

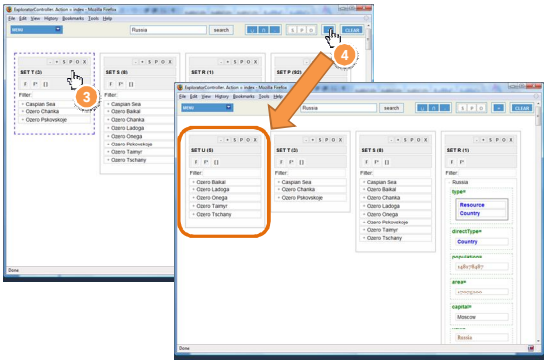

Figura 55 – Lagos contidos exclusivamente no território Russo.