5
Implementação da ferramenta proposta

O Excelplorator foi desenvolvido através da implementação de diversos novos conceitos no Explorator. Nesse capítulo serão discutidos alguns aspectos técnicos do Excelplorator e das modificações realizadas. Na primeira parte do capítulo é apresentada a arquitetura utilizada. Como todo o projeto foi desenvolvido seguindo a arquitetura e a forma de implementação utilizada no Explorator, esses detalhes técnicos serão brevemente discutidos, considerando que maiores detalhes podem ser consultados na dissertação de mestrado de Samur Araujo (PUC-Rio, 2009). A segunda parte do capítulo apresenta o novo modelo, que é a grande mudança estrutural realizada para possibilitar o desenvolvimento do Excelplorator. Finalmente, na última parte do capítulo, é realizada uma breve discussão técnica sobre a implementação das principais novas funcionalidades: a parametrização, as operações dinâmicas, a nova organização da área de trabalho, o sistema de autenticação e o compartilhamento de soluções.

Para facilitar o entendimento das funcionalidades e das soluções técnicas das implementações, serão utilizados exemplos similares aos dos capítulos anteriores.

5.1
A arquitetura utilizada

Como já foi mencionado anteriormente, o Excelplorator utiliza a arquitetura proposta pelo Explorator.

Toda a ferramenta foi desenvolvida em Ruby, uma linguagem interpretada e dinâmica, que segue o paradigma de orientação a objetos [Booch et al, 2007]. Para auxiliar e agilizar o desenvolvimento, foi utilizado o meta-framework Ruby-on-Rails[14], que utiliza o paradigma MVC [Leff et al, 2001]. Ruby-on-Rails é considerado um meta-framework por ser na realidade uma composição de

[14] Ruby on Rails - http://rubyonrails.org/

diversos outros frameworks. O paradigma MVC (model-view-controller) define a separação da aplicação em três camadas fundamentais: o modelo, responsável pela lógica de negócio; o controle, camada que responde as requisições, trata os dados e acessa a camada de modelo e a visualização, camada responsável pela exibição dos resultados para o usuário da aplicação.

Na parte de persistência, foi utilizado um framework chamado ActiveRDF, já mencionado no terceiro capítulo deste trabalho. O ActiveRDF provê uma camada de acesso aos dados representados em RDF. Ele suporta diversos tipos de armazenagens, como SPARQL endpoints ou bancos de dados RDF, como o Sesame[15]. Como Ruby é uma linguagem dinâmica, o framework consegue criar classes e métodos que refletem as classes e atributos existentes no modelo RDFS das bases acessadas, abstraindo de certa forma o modelo RDF em que os dados estão representados. Com isso o desenvolvimento pode ser feito de forma mais ágil. As triplas acessadas e criadas pelo Excelplorator são armazenadas através do ActiveRDF em uma base utilizando o banco de dados RDF Sesame.

A interface do Excelplorator, assim como a do Explorator, possui diversos efeitos visuais que visam facilitar a experiência do usuário. Um dos exemplos é a possibilidade de arrastar e reposicionar os conjuntos de recursos. Essa funcionalidade já existia no Explorator e foi complementada no Excelplorator com a adição do *drag n´drop*, realizado quando se deseja substituir no ambiente de desenvolvimento o valor de um parâmetro (Figura 46 e 49). Esses efeitos visuais foram adicionados a interface da ferramenta com o auxílio das bibliotecas Javascript[16] Scriptaculous[17] e Prototype[18].

Outra característica importante da interface da ferramenta desenvolvida é a utilização de requisições e atualizações da interface via Ajax, visando uma melhor interação com o usuário. A grande maioria das funcionalidades, como criação e edição de conjuntos, é realizada sem que a página precise ser recarregada. Toda a parte de navegação na aplicação gerada através do Excelplorator também é realizada via Ajax.

[15] http://www.openrdf.org/

[16] http://en.wikipedia.org/wiki/JavaScript

[17] http://script.aculo.us/

[18] http://www.prototypejs.org/

5.2 O novo modelo

A grande modificação estrutural realizada foi a alteração do modelo do Explorator, para que desse suporte às novas funcionalidades que foram implementadas. As diversas consultas montadas pelos usuários são armazenadas em RDF em uma base interna do Excelplorator. Para isso foi definida uma ontologia na qual todos dados de domínio são representados. Dessa forma eles podem ser persistidos para que depois sejam reutilizados, alterados e compartilhados.

É importante observar que qualquer ferramenta que conheça a nova ontologia definida pode gerar ou interpretar aplicações geradas com o Excelplorator.

O modelo utilizado pelo Explorator (Figura 87) é bastante simples e por isso não oferecia suporte para as novas funcionalidades. Ele é basicamente formado de duas classes, ResourceSet e Application. Cada instância de Application possui zero ou mais instâncias de ResourceSet. Cada ResourceSet possui uma instância de SemanticExpression. A classe SemanticExpression é de infra-estrutura, atuando como uma camada entre as classes de modelo e as bases RDF. Ela possui uma API que recebe requisições, acessa as bases habilitadas através de consultas utilizando a classe Query da API do ActiveRDF e retorna as triplas desejadas.

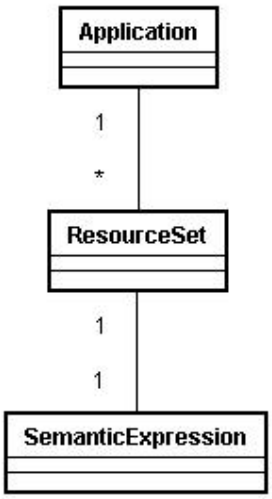

Figura 87 – Modelo do Explorator.

O modelo utilizado pelo Excelplorator é bem diferente e um pouco mais complexo. O aumento de complexidade foi necessário devido à necessidade de alterar a definição dos conjuntos de recursos para que as consultas possam ser generalizadas. Todas as classes de modelo do Excelplorator herdam de RDFS::Resource, classe do ActiveRDF, framework que cuida da persistência das instâncias na base RDF. Os Atributos herdados de RDFS:Resource são uri e class_uri. O primeiro define a URI do recurso, uma espécie de identificador único na base RDF interna do Excelplorator. O segundo define a URI da classe que recurso pertence.

O modelo do Excelplorator possui as seguintes classes listadas abaixo, que se relacionam como é demonstrado no diagrama (Figura 88):

- Application
- Workbench
- ResourceSet
- Operation
- Operand
- Transducer
- Parameter
- ParameterLink
- View

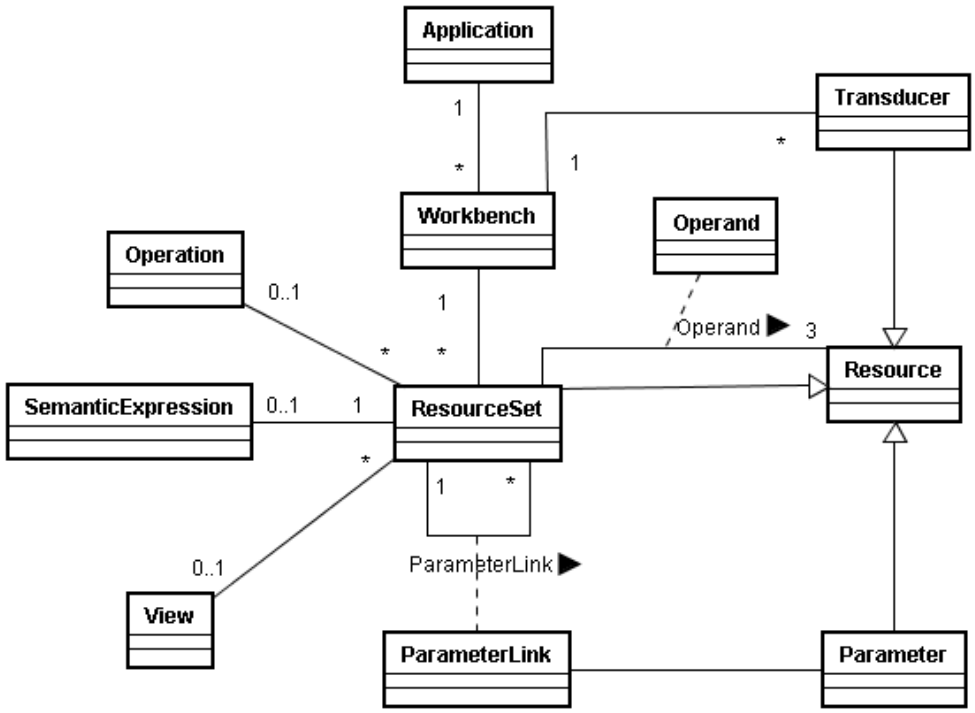

Figura 88 – Modelo do Excelplorator

5.2.1 A classe Application

A classe Application representa uma aplicação construída no Excelplorator. Para cada usuário que acessa o sistema, uma nova aplicação é instanciada com o nome padrão: "Default". Sempre que um usuário está utilizando o sistema existe uma aplicação ativa. Usuários podem persistir aplicações e carregar aplicações previamente persistidas a qualquer momento. No momento da persistência, é possível escolher um novo nome para a aplicação. Além de um nome, aplicações possuem um dono, um conjunto de operações habilitadas e um conjunto de workbenchs. As semânticas dos atributos da classe Application serão explicadas nos próximos parágrafos.

O dono de uma aplicação é definido quando um usuário logado no sistema persiste a aplicação ativa. Ela é automaticamente associada a um identificador único do usuário logado e ao nome do usuário fornecido pelo sistema de autenticação. O login no sistema e a obtenção de dados do usuário são realizados através do protocolo de identificação open-id [Recordon et al, 2006], que foi discutido no Cap. 4.3.5.1.

A aplicação também possui uma coleção de instâncias de recursos da classe Operation, que representa as operações que estão habilitadas nessa aplicação. É importante frisar que a coleção não tem semântica de posse, ela representa somente quais operações estão em uso em certo momento pela aplicação. Em qualquer momento do uso de uma aplicação, um usuário pode associar ou desassociar operações da aplicação ativa. Com isso ele estará habilitando e desabilitando o uso dessas operações na aplicação.

O último atributo da classe Application é uma coleção de recursos da classe Workbench. Essa coleção representa os workbenchs que compõem a aplicação. Toda aplicação possui no mínimo um workbench. O usuário pode criar, editar e apagar workbenchs das aplicações.

5.2.2 A classe Workbench

O recurso da classe Workbench representa um conjunto de consultas que pretende responder a uma pergunta específica, por exemplo, quais são as publicações científicas de um determinado pesquisador.

Recursos da classe Workbench podem ser criados e apagados pelo usuário e sempre pertence a uma aplicação. Cada aplicação possui sempre no mínimo um workbench e assim como sempre existe uma aplicação ativa para cada usuário utilizando o sistema, sempre existe também um workbench ativo. Recursos da classe Workbench possuem nome, descrição, um indicador que define se o workbench pode ser copiado por outros usuários, uma coleção de transdutores e uma coleção de conjuntos de triplas.

Quando um novo workbench é criado pelo usuário, ele recebe o nome “new workbench”, uma descrição nula e não está compartilhado com outros usuários. Esses três atributos são utilizados principalmente para o compartilhamento e podem ser alterados a qualquer momento pelo dono da aplicação.

Um workbench também possui uma coleção de transdutores e uma coleção de conjuntos de triplas. A primeira coleção é de recursos da classe Transducer e a segunda é de recursos da classe ResourceSet. Essas duas coleções de recursos e as relações entre elas definem as consultas realizadas no workbench. Tanto

transdutores quanto conjuntos de triplas podem ser criados pelo usuário e sempre pertencem ao workbench ativo no momento da criação dos mesmos.

5.2.3 A classe ResourceSet

A classe ResourceSet representa um conjunto de triplas RDF. As triplas de um resourceset, instância de ResourceSet, são um subconjunto das triplas existentes nas bases RDF habilitadas no Excelplorator. Instâncias da classe ResourceSet são criadas pelo usuário e as suas triplas são definidas de duas formas: ou através de uma expressão que define uma consulta SPARQL às bases habilitadas ou através de uma operação em cima de outros recursos. Um resourceset criado diretamente com uma expressão que define uma consulta SPARQL não pode ser modificado depois de sua criação. Já um resourceset criado a partir de uma operação em cima de outros recursos pode ser editado e modificado. Um resourceset desse tipo possui um operador e operandos. O operador é definido no momento de criação do resourceset e não pode ser modificado. Já os operandos são coleções de recursos e podem ser alterados. A coleção que define um operando pode possuir qualquer recurso disponível nas bases habilitadas, inclusive recursos das classes definidas pelo Excelplorator. Instâncias de Parameter, Transducer e Operand possuem comportamento diferente quando utilizados como operandos de operações. Seus valores são substituídos respectivamente pelo valor do parâmetro, valor do transdutor e pelo conjunto de recursos do resourceset do operando.

Recursos da classe ResourceSet também possuem mais três atributos, um indicador booleano, um recurso da classe View e uma coleção de recursos da classe ParameterLink. O primeiro define se o resourceset deve se comportar como um dos conjuntos de triplas que funcionam como índice inicial (exibido quando o caso de uso é acessado) durante o compartilhamento fechado da aplicação. O segundo atributo define se a visualização do conjunto de recursos será customizada durante a utilização de uma aplicação gerada. Já o terceiro, a coleção de recursos da classe ParameterLink, define a dupla resourceset e parâmetro para qual o valor selecionado pelo usuário final da aplicação compartilhada de modo fechado deve ser enviado. Caso um resourceset possua um a instância de um

ParameterLink, quando um usuário escolher um dos seus recursos, a instância escolhida será passada como parâmetro para onde o link estiver apontando.

5.2.4 A classe Operation

A classe de domínio Operation representa uma operação sobre um ou mais conjuntos de triplas. O resultado da operação é sempre o conjunto de triplas de um resourceset. As operações podem ser criadas dinamicamente por usuários e compartilhadas. O código que define o comportamento da operação é um atributo da classe Operation, e deve ser um trecho de código na linguagem Ruby. Uma instância da classe Operation também possui um nome e um atributo com o símbolo da operação, que deve ser um caractere que represente a operação na ferramenta.

5.2.5 A classe Operand

A classe Operand existe para adicionar atributos no relacionamento entre um conjunto de triplas e seus operandos. Quando um conjunto de triplas tem como um de seus operandos outro conjunto de triplas, ele necessita de um atributo extra para indicar qual projeção deve ser considerar como seu parâmetro. A projeção é uma das posições das triplas: sujeito, predicado ou objeto. A classe Operand possui um atributo que adiciona esse meta-dado ao relacionamento. Por exemplo, considerando um conjunto de triplas A que é definido por uma operação SPO e possua como único parâmetro outro conjunto de triplas B para a posição do sujeito de A. Caso a projeção de B seja sujeito, serão utilizados os recursos que estão na posição do sujeito nas triplas de B, caso seja predicado, serão utilizados os recursos que estão na posição de predicado nas triplas de B e caso seja objeto, serão os recursos que estão na posição de objeto nas triplas de B. Portanto, é necessário guardar qual projeção de B será utilizada na operação.

5.2.6 A classe Parameter

A classe Parameter foi criada para facilitar a substituição de valores que compõem os operandos de um resourceset. Com essa substituição é possível a generalização de consultas, possibilitando o reúso. Um recurso da classe Parameter é criado quando o usuário seleciona um recurso de um dos operandos de um resourceset para ser parametrizado. Um parâmetro possui um nome, que é gerado pelo sistema no momento da sua criação, e um tipo de entrada de dados. Ambos podem ser editados pelo usuário.

5.2.7 A classe ParameterLink

Recursos da classe ParameterLink pertencem a um resourceset e são utilizados pelo compartilhamento fechado da aplicação. Quando um resourceset possui um ParameterLink, o recurso selecionado pelo usuário final da aplicação é enviado como valor de um parâmetro para um outro resourceset. Para isso, recursos da classe ParameterLink possuem uma instância da classe Parameter, a instância da classe ResourceSet a qual o parâmetro pertence e uma projeção (sujeito, predicado ou objeto). Quando um recurso R de um conjunto de triplas A é selecionado por um usuário na aplicação fechada, e exista alguma instância de ParameterLink de A que possua a projeção que R pertence, o valor do recurso R será enviado para ser avaliado na posição do parâmetro indicado pela instância do ParameterLink.

5.2.8 A classe Transducer

A classe de domínio Transducer representa uma entrada de dados realizada diretamente pelo usuário. Através de um recurso da classe Transducer, usuários podem entrar com dados que não fazem parte das bases RDF habilitadas. Esses dados podem ser utilizados pelas operações na formação de conjuntos de triplas. Tansdutores são relativamente simples, possuem um nome e um valor, ambos editáveis. A única forma de entrada de dados implementada no Excelplorator é via texto livre.

5.2.9 A classe View

A última classe do modelo proposto pelo Excelplorator é a View. Um recurso da classe View representa uma customização de interface, que pode ser utilizada nas aplicações fechadas. A classe possui dois atributos, um nome e o código fonte em HTML customizado. Usuários da ferramenta podem criar e alterar instâncias da classe View e dessa forma alterar a aparência e o comportamento da camada de apresentação.