# 4
# Modelo de Controle de Acesso no SHDM

## 4.1.
## O Método SHDM

SHDM (*Semantic Hypermedia Design Method*) é um método para o projeto de aplicações hipermídia para a *web* semântica [Lima, 2003]. O método SHDM é uma evolução do método OOHDM (*Object Oriented Hypermedia Design Method*) através da introdução de formalismos e primitivas da *Web* Semântica. Ele permite o projeto de aplicações hipermídia baseadas em ontologias, descritas através de metadados.

O método OOHDM é um método baseado em modelos que utiliza técnicas de orientação a objetos para o design de aplicações *web* [Schwabe e Rossi, 1998]. Devido ao fato do método SHDM ser uma evolução do OOHDM, ele manteve os mesmos fundamentos e a mesma estrutura de fases de modelagem do método original.

O método SHDM será apresentado neste capítulo de forma resumida com ênfase para as mudanças introduzidas neste trabalho, mais especificamente no Projeto Comportamental. Uma descrição mais detalhada do método SHDM pode ser obtida em trabalhos anteriores, tais como [Lima, 2003], [Szundy, 2004], [Nunes, 2005] e [Bomfim, 2011].



### 4.1.1.
### Etapas

O método SHDM é composto por seis atividades distintas essenciais para o desenvolvimento de aplicações hipermídia na *web* semântica, sendo estas: Levantamento de Requisitos, Modelagem de Domínio, Projeto Comportamental, Projeto Navegacional, Projeto de Interface e Implementação.

Cada etapa do método SHDM produz um ou mais modelos descritos em RDF que focam em aspectos específicos e distintos de uma aplicação hipermídia. Essas etapas não precisam ser executadas em uma ordem específica e apenas uma vez, elas podem ser executadas de forma iterativa e incremental. O projetista deve modelar cada aspecto da aplicação separadamente em cada etapa do método, porém os artefatos produzidos em

uma etapa podem sofrer mudanças estimuladas por requisitos identificados em outras etapas. Visto que todos os modelos são descritos em RDF, é possível que qualquer um deles sofra alterações pela inclusão de novas triplas, motivadas por etapas diferentes de suas etapas originais [Bomfim, 2011].

### 4.1.2. Levantamento de Requisitos

A etapa de Levantamento de Requisitos é responsável por identificar os atores e as tarefas a serem apoiadas pela aplicação. Nesta etapa são produzidas descrições de cenários e dos casos de uso, e Diagramas de Interação com o Usuário (*User Interaction Diagrams* – UIDs).

Um Diagrama de Interação do Usuário ou UID (*User Interation Diagram*) representa a interação entre o usuário e uma aplicação hipermídia. Este diagrama descreve somente a troca de informações entre o usuário e a aplicação, sem considerar aspectos específicos da interface com o usuário nem da navegação [Vilain, 2002]. Um UID é composto por um conjunto de estados conectados através de transições. Os estados representam as informações que são trocadas entre o usuário e a aplicação, enquanto que as transições são responsáveis pela troca do foco da interação de um estado para outro.

A Figura 12 apresenta um exemplo de um UID que representa a interação entre o usuário e o sistema durante a tarefa *Seleção de um CD a partir de um título*. Para mais detalhes sobre a notação deste diagrama, ver em [Vilain, 2002].

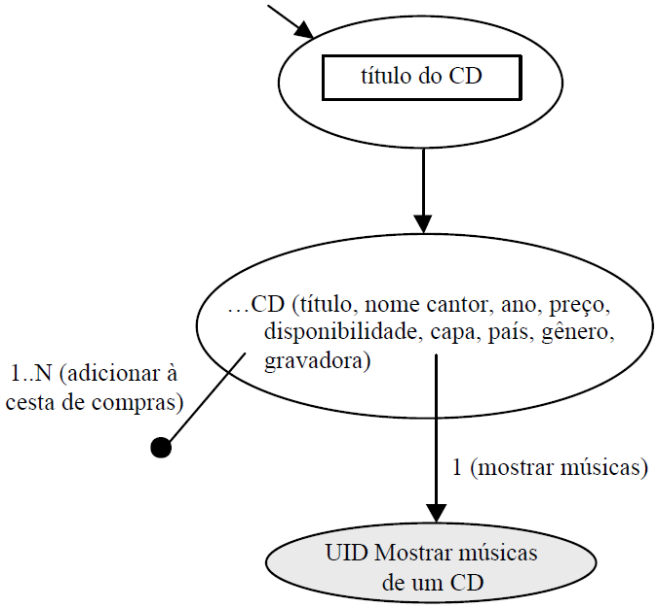


Figura 12 – Exemplo de UID como apresentado por [Vilain, 2002]

### 4.1.3.
### Modelagem de Domínio

A fase da Modelagem de Domínio descreve os dados do domínio de uma aplicação na *web* semântica e os seus relacionamentos. Esta fase produz o modelo de domínio, que descreve os dados reais do domínio de uma aplicação compostos por recursos, classes e propriedades RDF. Portanto, a base estrutural do modelo de domínio é o modelo RDF e é possível que se utilize qualquer ontologia definida para a *web* semântica (em OWL ou RDFS).

O modelo de domínio é composto por um conjunto de ontologias de domínio e mais duas primitivas introduzidas no trabalho de Bomfim (2011): os *namespaces* e os repositórios de dados da *web* semântica ou *datasets*.

Os *namespaces* são usados para identificar quais vocabulários estão sendo usados na ontologia de domínio. Eles são formados por um prefixo e uma URI. Os *datasets* são repositórios de dados da *web* semântica disponibilizados pelo projeto *Linking Open Data*[23], tais como o DBPedia[24], Geonames[25], MusicBrainz[26], LinkedMDB[27], etc. Tais repositórios são úteis para complementar as instâncias do domínio da aplicação. Desta maneira, uma aplicação hipermídia pode ter a combinação de dados locais e dados remotos em seu modelo de domínio.

As ontologias de domínio devem ser criadas de acordo com as práticas do *Linked Data* [Bizer et al., 2009], como fazer a reutilização de termos presentes em vocabulários bem conhecidos pela comunidade da *web* semântica.

### 4.1.4.
### Projeto Navegacional

O objetivo da atividade de Projeto Navegacional é definir o modelo navegacional no qual representa uma visão navegacional sobre os dados do modelo de domínio, definindo quais informações poderão ser acessadas pelos usuários da aplicação e como estas informações poderão ser exploradas [Szundy, 2004].

[23] http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData

[24] http://dbpedia.org

[25] http://www.geonames.org/ontology

[26] http://musicbrainz.org

[27] http://linkedmdb.org/

A partir do trabalho de Bomfim (2011), a tarefa de mapeamento de classes navegacionais a partir do esquema conceitual de classes que resultava no esquema de classes navegacionais deixou de existir, pois gerava um novo modelo de dados RDF que era desnecessário. Ao invés disso, as instâncias do modelo de domínio foram enriquecidas para conter as descrições dos atributos navegacionais e o conceito de nó navegacional passou a ser entendido como uma instancia do domínio (recurso RDF) acessado a partir de um contexto navegacional.

Portanto, a fase de Projeto Navegacional produz um modelo navegacional, que é um sub-grafo do modelo de domínio enriquecido com atributos navegacionais e produz também um modelo de contextos navegacionais. Este último tem como primitivas os contextos de navegação, as estruturas de acesso (índices) e os *landmarks* da aplicação.

O modelo de contextos navegacionais especifica a estrutura navegacional de uma aplicação *web* por meio de contextos navegacionais. Contextos navegacionais são conjuntos de nós navegacionais que são acessados pela aplicação. Exemplos de contextos navegacionais são todos os filmes; filmes por gênero; CDs gravados por um artista; artigos associados a um revisor; etc. A Figura 13 mostra um exemplo de modelo de contextos navegacionais. Para mais detalhes sobre a notação dos esquemas de contextos navegacionais, ver em [Lima, 2003].

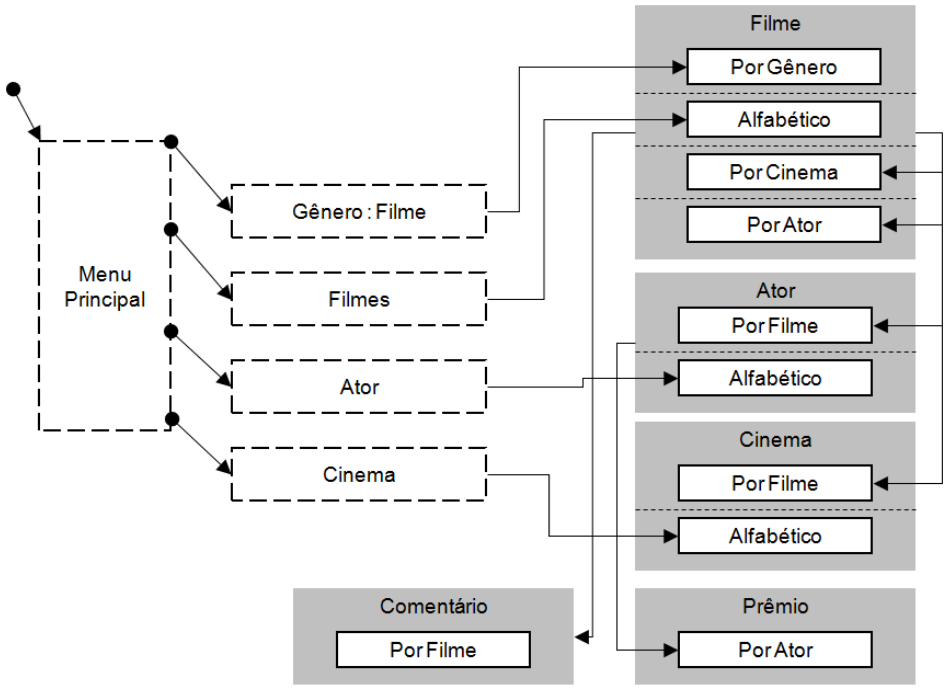


Figura 13 – Exemplo de modelo de contextos navegacionais

### 4.1.5.
### Projeto Comportamental

Na atividade de Projeto Comportamental ocorre a descrição da lógica de negócio da aplicação *web*, através da especificação das operações da aplicação. A partir deste trabalho, o Projeto Comportamental passou a ter também a especificação de controle de acesso da aplicação. Dois modelos serão produzidos nesta fase: o modelo de operações e o modelo de controle de acesso baseado em papel, apresentado este último no capitulo 3.

O modelo de operações contém a representação de todas as operações do domínio da aplicação identificadas no Levantamento de Requisitos. O papel principal de uma operação é causar transformações no estado da aplicação. Porém, existem operações que não causam mudança no estado corrente de um objeto, como por exemplo, a operação “enviar e-mail” ou “imprimir matéria”.

O modelo de operações foi originalmente proposto por Santos (2010), e teve algumas alterações em seu vocabulário feitas por Bomfim (2011), mas mantendo a mesma semântica da proposta original. A Figura 14 mostra o vocabulário do modelo de operações.

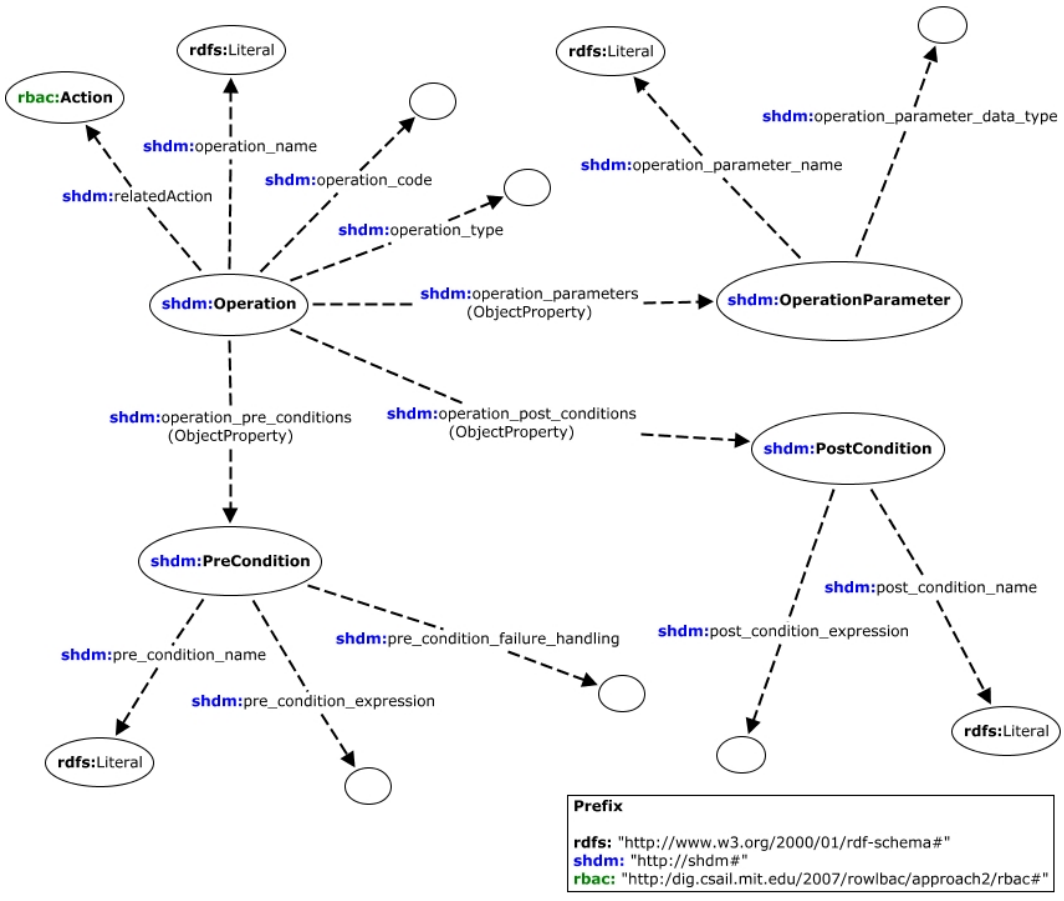


Figura 14 – Vocabulário do modelo de operações

A seguir segue a descrição de cada primitiva do modelo de operações, destacando-se as mudanças ocasionadas por este trabalho.

Uma operação é especificada como uma instância da classe shdm:Operation. A propriedade shdm:operation_name define o nome da operação. A propriedade shdm:operation_code contém a descrição do código da operação. A linguagem utilizada para a definição do código da operação (shdm:operation_code) depende do ambiente de implementação da aplicação. O método SHDM não prescreve o uso de nenhuma linguagem específica. Normalmente utiliza-se a mesma linguagem de programação utilizada na implementação da aplicação. No ambiente Synth, apresentado no capítulo 5, os códigos das operações são descritos na linguagem Ruby[28].

A propriedade shdm:operation_type define o tipo de operação. Existem dois tipos de operações: as operações internas e as operações externas. As operações internas descrevem as regras de negócio da aplicação. Elas não são visíveis às entidades externas (usuários ou sistemas) à aplicação, portanto, não podem ser invocadas por estas. No entanto, elas podem ser invocadas por outras operações e pelo modelo de interface. As operações externas podem ser invocadas por entidades externas e funcionam como um canal de comunicação entre tais entidades e a aplicação. Elas são invocadas por meio do envio de requisições do protocolo HTTP (Hypertext Transfer Protocol) ao servidor e produzem como resultado a renderização de uma interface visual ou o redirecionamento a outra URL.

A propriedade shdm:relatedAction foi adicionada ao modelo de operações para representar que um recurso de shdm:Operation tem uma rbac:Action correspondente. Esta relação define que o recurso da classe shdm:Operation está sendo controlado pela aplicação através de primitivas do modelo de controle de acesso RBAC, visto no capitulo 3.

A classe shdm:OperationParameter representa os parâmetros de entrada de uma operação. Os parâmetros da operação são usados para identificar os objetos (rbac:Object) do modelo de controle de acesso RBAC, ou seja, quando uma operação (shdm:Operation) está sendo controlada, os parâmetros desta operação representam os objetos do modelo de controle de acesso. Uma shdm:Operation pode possuir zero ou mais parâmetros. No caso da operação possuir mais de um parâmetro, a verificação da permissão na lista de controle de

[28] http://www.ruby-lang.org

acesso (ACL) retornará verdadeira, ou seja, tem permissão, quando pelo menos um parâmetro for igual ao recurso que representa o objeto na ACL.

A propriedade shdm:operation_parameter_name identifica o nome do parâmetro, enquanto que a propriedade shdm:operation_parameter_data_type representa o tipo de dado do parâmetro. Existem sete tipos de dados definidos no modelo de operações: String, Integer, Float, Date, Hash, Array, List. Para que os objetos (recursos RDF da classe rbac:Object) pudessem ser representados como parâmetro, um novo tipo de dado foi acrescentado no modelo: RDFS::Resource. Nas operações externas, os parâmetros da operação são passados pela URL.

Uma operação pode ter pré-condições (shdm:PreCondition) e pós-condições (shdm:PostCondition). As pré-condições são avaliadas antes da execução da operação. A operação só será executada se a pré-condição for verdadeira. Se for falso, um código de tratamento de falha (shdm:pre_condition_failure_handling) é executado. A propriedade shdm:pre_condition_expression contém o código da pré-condição e a propriedade shdm:pre_condidion_name define o nome da pré-condição.

Duas pré-condições devem ser definidas automaticamente pelo ambiente de execução sempre que uma operação é definida como protegida pelo modelo de controle de acesso:

- Pré-condição que faz a checagem se o usuário está autenticado na aplicação. Caso não esteja, o código de tratamento de falhas redireciona o usuário para a tela de login.
- Pré-condição que verifica na lista de controle de acesso (ACL) se o usuário tem permissão de executar a operação. Caso retorne falso, o código de tratamento de falhas pode mostrar uma mensagem de erro como “Access Denied”.

Por fim, as pós-condições (shdm:PostCondition) são avaliadas após a execução da operação. A propriedade shdm:post_condition_name define o nome da pós-condição, enquanto que a propriedade shdm:post_condition_expression especifica o código da pré-condição. Não existe tratamento de falha para as pós-condições.

A semântica de navegação pode ser especificada como uma operação pré-definida do modelo de operações do SHDM. Isto porque a semântica de navegação é bem definida, isto é, ela pode ser descrita por um conjunto de passos processáveis pela aplicação, além dela ser independente do modelo de domínio da aplicação. Portanto, a operação de navegação, isto é, percorrer de

um nó origem até um nó destino é descrita usando o vocabulário do modelo de operações para dar semântica ao modelo navegacional.

### 4.1.6. Projeto de Interface

A fase de Projeto de Interface tem o objetivo de construir o modelo de interfaces da aplicação, que representa os elementos da aplicação percebidos pelo usuário e as interações do usuário com esses elementos. O modelo de interfaces foi proposto originalmente por [Moura, 2004] e estendido no trabalho de [Luna, 2010]. A descrição desta etapa não será apresentada neste trabalho.

### 4.1.7. Implementação

Por fim, a etapa de Implementação tem como objetivo transformar todos os modelos definidos nas etapas anteriores do método SHDM em uma aplicação hipermídia executável. Os modelos são mapeados e implementados em um determinado ambiente de desenvolvimento.