
Ministério da
Ciência, Tecnologia
e Inovação


# MODELAGEM DE METADADOS DE OBSERVAÇÕES ESPAÇO-TEMPORAIS DIRECIONADA À INTEROPERABILIDADE

Trabalho da disciplina Introdução ao Geoprocessamento, do curso de mestrado em Computação Aplicada

Diego Benincasa Fernandes Cavalcanti de Almeida

INPE

São José dos Campos

2014

**SUMÁRIO**

# I INTRODUÇÃO

## 1.1 Descrição do problema

Nos dias atuais, é cada vez maior o número de aplicações e projetos que se utilizam de dados georreferenciados. A informação geográfica vem se tornando um importante componente de análise em diversos setores, tanto em estudos da Terra quanto em tarefas de planejamento e gerência. Aliado à alta velocidade de evolução tecnológica do mundo, conjuntos de ferramentas capazes de tratar dados espaciais são constantemente desenvolvidos e melhorados, no intuito de otimizar os processos que levam a tomadas de decisões.

Uma das análises espaciais que constantemente são realizadas é a comparação de dados em momentos distintos no tempo. Isto é utilizado, por exemplo, para análises de séries temporais de chuvas em uma determinada região, ou para avaliação da expansão de áreas de desmatamento. Nestes tipos de estudos, a componente temporal do dado é essencial, haja vista que define as nuances de um dado ou, em outras palavras, as suas instâncias. Diz-se que cada instância de um dado é uma observação do mesmo, com um momento definido em que ocorreu e um conjunto de propriedades associadas relativas a este instante.

Observações podem estar computacionalmente armazenadas em diversos tipos de estruturas, cada uma delas tendo seu próprio modelo de armazenamento associado, codificado segundo um destes tipos (um formato de arquivo ou de banco de dados). A escolha da estrutura adequada da fonte de dados é critério do produtor ou do gerente das observações, mas uma vez definida deve se manter coerente, ou seja, todos os dados devem estar corretamente referenciados a um modelo de organização, definido pela estrutura da fonte.

Apesar de existirem diversos modelos e padrões de armazenamento e representação de dados espaciais, não existe ainda um modelo único para descrever aqueles que são variantes ao longo do tempo. Sendo assim, torna-se difícil garantir a interoperabilidade entre fontes distintas de dados espaço-temporais, uma vez que não compartilham informações segundo um padrão de comunicação. De modo corriqueiro, diz-se que estas fontes “não conversam”.

## 1.2 Motivação

Conforme anteriormente relatado, há uma lacuna na modelagem de dados espaço-temporais no que tange à interoperabilidade. Ao mesmo tempo, é facilmente vislumbrável a gama de aplicações de tais dados que, assim como ocorre com os dados espaciais estáticos, devem ser

intercambiáveis entre sistemas e fontes que se utilizam da componente temporal para descrever seus elementos geográficos. Isto promove a redução, ou mesmo eliminação, principalmente da redundância e da descontinuidade de informações espaciais, entre outros fatores. Assim, nota-se a relevância de trabalhos que busquem, se não criar um modelo de armazenamento padrão do dado, ao menos estabelecer um modelo para descrição da estrutura deste armazenamento, que esclareça a clientes deste de que forma e quais as informações estão relacionadas no tempo.

# II OBJETIVOS

## 2.1 Objetivo geral

O principal propósito deste trabalho é sugerir um modelo para estruturação dos metadados de fontes de dados espaciais, particularmente para descrever a relação temporal entre dados armazenados nestas fontes. Para tal, usar-se-á o modelo estabelecido pelo *Resource Description Framework* (RDF), focado em recursos estabelecidos segundo os padrões de interoperabilidade da OGC.

## 2.2 Objetivos específicos

Para se atingir o objetivo geral, etapas intermediárias serão sequencialmente cumpridas, a saber:

1. Definição do modelo abstrato de relacionamento entre dados espaço-temporais.
2. Tradução do modelo abstrato para um modelo XML.
3. Conversão lógica do modelo XML para RDF/XML.
4. Análise do modelo definido na etapa anterior quanto à adequabilidade ao caso real.

# III FUNDAMENTOS

## 3.1 XML e XSD

O conceito de metadado não é novo, e vários tipos de armazenamento de dados fazem uso deste recurso para descrever suas estruturas internas. Costumeiramente, os metadados constituem-se em um conjunto de informações extras sobre uma informação principal, que a descreve melhor e facilitam o seu uso para um determinado propósito. Porém, para que os metadados apresentem um sentido e possam de fato serem aproveitados, é necessário que estejam organizados ou, em termos de sistemas de computadores, coerentes com um modelo de estruturação. Assim, antes de se prover informações complementares sobre dados, precisa-se definir o modelo que seguirão.

Modelos de representação de metadados não são únicos, e variam de acordo com o conjunto de dados que se deseja descrever. Desta forma, surgiram ao longo do tempo diversos mecanismos que pudessem facilitar a tarefa de modelar, como algumas linguagens peculiares de elaboração de documentos de texto. Um exemplo destas linguagens é o *Extensible Markup Language* (XML), cuja particularidade é seu recurso de encapsular informações em diferentes contêineres (*tags*), que nada mais são do que blocos de texto. Segundo Bray et al. (2006), um arquivo escrito em linguagem XML é um arquivo de texto simples, cuja formatação segue um modelo de grupos e subgrupos de informações, semelhante a capítulos e tópicos de um livro. As definições, relações e hierarquias destes contêineres de informação constituem o que se denomina esquema, que é estabelecido em um segundo arquivo, também escrito em linguagem XML, denominado *XML Schema Definition* (XSD). É o XSD que informa, para o usuário do arquivo XML, qual é a sua organização interna.

```
<recipe>

    <title>Peanut-butter On A Spoon</title>

    <ingredientlist>
        <ingredient>Peanut-butter</ingredient>
    </ingredientlist>

    <preparation>
        Stick a spoon in a jar of peanut-butter,
        scoop and pull out a big glob of peanut-butter.
    </preparation>

</recipe>
```

Figura 3.1 – Exemplo demonstrativo de arquivo escrito em linguagem XML.

(Fonte: Wikipedia)

Os benefícios no uso de arquivos XML estão em sua extensibilidade e flexibilidade, bem como na clara facilidade de criação, visualização e manipulação, notadamente por se tratarem, em última análise, de arquivos de texto puro. Além disto, sua lógica interna é igualmente simples e de fácil entendimento, sendo a dita organização estabelecida pelo arquivo XSD. Para ler corretamente os dados armazenados em arquivos XML, existem atualmente diversas ferramentas que, além desta função, validam o mesmo de acordo com seu esquema XSD, bem como interpretam o XML e recuperam os diversos dados que armazena segundo a organização que seu criador projetou. Estas ferramentas são denominadas *parsers*, e a leitura do arquivo, interpretando cada contêiner corretamente, é chamada de *parsing*.

No campo das informações geoespaciais, é relevante ressaltar que o XML é adotado como padrão para descrição de informações e serviços *web* geoespaciais (ISO, 2014). Além disto, a *Open Geospatial Consortium* (OGC), organização mundial sem fins lucrativos voltada à criação e manutenção de padrões de interoperabilidade entre sistemas e bases de dados espaciais, adota o XML no formato de arquivo padrão que define para intercâmbio de informações espaciais, o *Geography Markup Language* (GML) (OGC, 2012).

## 3.2 *RDF*

O XML é prático para diversas aplicações, e funciona a contento quando se procura intercambiar dados entre sistemas diferentes. Um exemplo, já citado, é o formato GML da OGC, bem como todos os sistemas que fazem uso deste padrão de arquivo. Contudo, é apenas uma linguagem de estruturação de dados que, por ser extensível, permite adaptações e evoluções. Uma das possíveis melhorias que pode ser realizada na linguagem XML diz respeito a permitir que ela possa descrever metadados de recursos. Um recurso é, basicamente, uma fonte de dados que é passível de ser descrita por um conjunto de atributos informativos relevantes (os ditos metadados) e que está disponível em algum local segundo uma identificação única.

Quando se analisa o conceito de interoperabilidade, é imediato inferir que os recursos disponíveis no mundo sejam deveras heterogêneos. No âmbito da informação geoespacial, existem, e sempre irão existir, diversos produtores de dados dos mais diversos domínios, o que enfatiza a necessidade de se efetuar esforços para padronização daquilo que de fato produzem. Estes esforços, notadamente, se veem centralizados nas normas estabelecidas pela OGC. Porém, embora existam documentações suficientes para padronizar o armazenamento e a troca de dados espaciais entre sistemas, pouco se avançou na interoperabilidade destes mesmos dados quando eles são variantes no tempo (dados espaço-temporais). A OGC padronizou o armazenamento de dados de observações

espaço-temporais em arquivos XML (COX, 2011), mas não possui algo semelhante para outros recursos e seus metadados, bem como para o intercâmbio.

Para a finalidade desejada, existe o modelo *Resource Description* Framework (RDF), definido pela OGC como padrão para descrição de metadados sobre recursos (OGC, 2012). É uma modelagem conceitual abstrata, cuja concretização pode se dar por diversos meios, sendo um deles a escrita de arquivos XML. Quando isto acontece, é costume referir-se ao dito arquivo como sendo de formato RDF/XML (neste trabalho, todas as referências futuras a arquivos deste formato serão feitas apenas por “RDF”). Sua principal diferença em relação ao XML tradicional são os contêineres de informação: no RDF, cada um deles representa um conjunto de descrições complementares atreladas a um determinado recurso. Seu uso proposto quando de sua criação foi para integração entre recursos na *web*, mas não há impedimentos para utilizá-lo com recursos locais *off-line* (arquivos de dados espaciais, por exemplo), já que a referência ao recurso, disponível ou não na *web*, deve ser um identificador único do mesmo, normalmente seu *Uniform Resource Locator* (URL) ou, de forma geral, seu *Uniform Resource Identifier* (URI)..

Na estrutura RDF, cada declaração de metadado é composta de três elementos principais: um sujeito (o qual deseja-se atribuir informações), um objeto (a informação que será atribuída ao sujeito) e um predicado (a relação de atribuição). Uma declaração desta forma é chamada de *tripla*. Esta definição não é diferente do que existe no XML simples, porém a diferença reside exatamente nos domínios de cada uma das três funções discriminadas: enquanto no XML simples todos são termos literais definidos pelo produtor do arquivo, no RDF podem ser URI's de recursos (ao menos o sujeito é).

A vantagem citada anteriormente, relativa à capacidade de descrever recursos, traz consigo mais benefícios. Conforme visto, os metadados de um determinado recurso, descritos no arquivo RDF, podem igualmente ser recursos. Assim, tudo em um RDF pode igualmente possuir um conjunto de metadados. Um exemplo disto pode ser visto na Fig. 3.2. Seja um banco de dados que possua uma tabela de cadastro de pessoas físicas, com seus números de identidade, seus nomes e seus endereços. Os nomes dos cadastrados podem seguir um determinado formato (no exemplo, constitui-se dos campos nome, sobrenome e CPF), mas esta organização poderá estar definida em outro recurso (arquivo RDF, por exemplo).

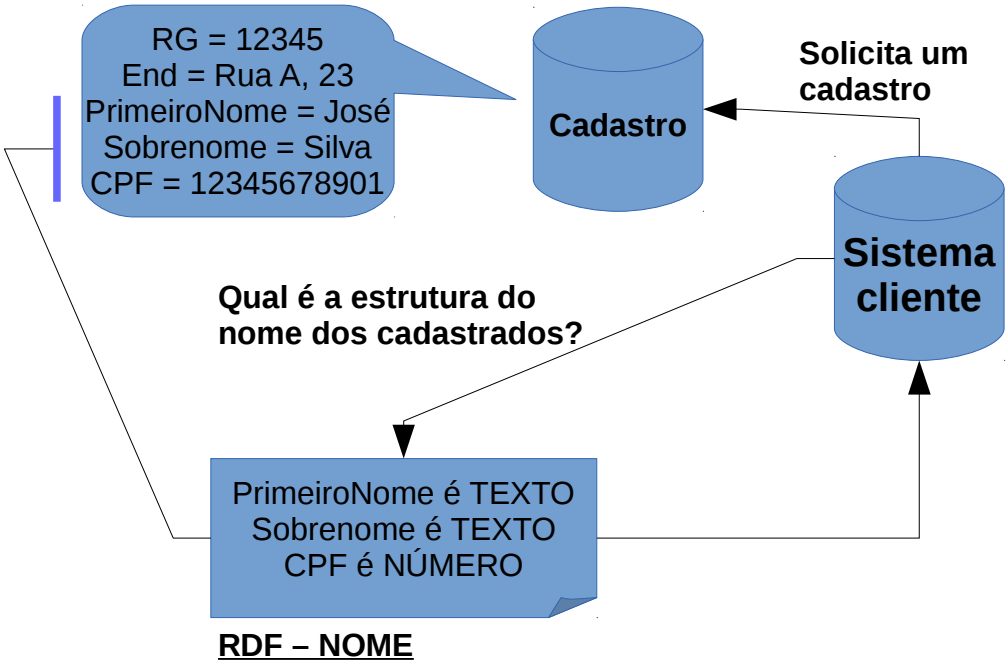


Figura 3.2 – Um exemplo de uso do arquivo RDF para descrever metadados de recursos.

(Fonte: o autor)

A interligação entre metadados cria uma nova relação entre significados, que ao ser analisada num contexto interoperacional remete a duas definições: *Semantic Web* e *Linked Data*. A primeira contempla um conjunto de esforços visando definir computacionalmente, de forma padronizada, conceitos do mundo real, numa espécie de "dicionário padrão". A segunda relaciona-se diretamente à ligação entre recursos distintos que, quando estruturados de acordo com semânticas conhecidas ou passíveis de serem entendidas, permitem que sistemas conversem entre si, sem intervenção humana para atribuir significados aos elementos relacionados.

Para criar padrões de significados (ou "dicionários"), existem diversos esquemas ontológicos já propostos, cada um visando definir elementos de um contexto específico. Um determinado esquema poderá, por exemplo, descrever o que são séries de televisão, através de um conjunto de atributos inerentes a elas. Assim, todo RDF que possua informações sobre este tipo de programa televisivo poderá fazer um *link* para o conjunto de definições que constroem o conceito de série de TV, retirando do sistema consumidor deste dado a necessidade de saber o significado disto, e mais profundamente dos elementos construtores do significado. Estes arquétipos da estrutura do RDF que definem significados constituem-se em um modelo chamado *RDF Schema* (RDFs), escrito também em XML, a exemplo do RDF.

Uma analogia que pode ser feita para entendimento dos princípios de *Semantic Web* e *Linked Data* é no diálogo entre pessoas nascidas em países distintos, por exemplo Brasil e Suécia. Para que haja um diálogo compreensível entre as partes, é preciso que uma delas compreenda o que a outra fala. Esta compreensão é composta por duas etapas: saber qual é o idioma nativo do interlocutor, e saber se comunicar neste idioma. O brasileiro, assim, precisa ter a informação de que seu interlocutor fala sueco para utilizar o dicionário correto na tradução dos termos falados, ao mesmo tempo que neste dicionário deve haver as palavras utilizadas. Palavras em idiomas diferentes, de fato, descrevem a mesma coisa, o mesmo elemento do mundo real. Ou seja, possuem o mesmo significado. Uma vez que o brasileiro consegue traduzir a palavra falada segundo o dicionário correto para o idioma, a compreensão é concluída. Desta feita, por analogia, temos que a descoberta ou obtenção do idioma falado pelo interlocutor está para os princípios de *Linked Data*, assim como o significado intrínseco às palavras utilizadas no referido idioma está para os conceitos de *Semantic Web*.

## 3.3 Observações espaço-temporais

Todo dado geoespacial que possui informação temporal associada é denominado dado espaço-temporal. Dispondo-se de um conjunto de dados espaço-temporais, uma instância destes dados (ou seja, o subconjunto dos mesmos que existe num tempo considerado) é dita ser uma observação. Assim, de modo sucinto e de acordo com Sinton apud Ferreira et al. (2013, p. 11), toda observação deve ser composta de três elementos principais: espaço (referente aos locais de cobertura do dado), tempo (relativo ao instante considerado) e tema (inerente ao contexto estudado ou analisado, ao fenômeno ou entidade do mundo real sendo observado).

Hoje, conforme já eslarecido, os recursos que proveem dados espaço-temporais dispõem de padrões de estruturação dos mesmos definidos pela OGC, bem como para interoperabilidade entre sistemas. Contudo, não existe a mesma disponibilidade no que tange às observações. Um início de estudo neste sentido encontra-se no trabalho de Ferreira et at. (2012), que propôs uma modelagem de metadados de observações que resultem em trajetórias, ou seja, em objetos que se deslocam no espaço ao longo do tempo. Porém, este foi um início de um estudo mais amplo que ainda carece de conclusão.

# IV DEFINIÇÃO DA PROPOSTA DA ESTRUTURA DE METADADOS

## 4.1 Modelagem do RDFs de observações espaço-temporais

Todo dado espacial variante no tempo, armazenado segundo alguma estrutura, possui basicamente três informações que o determinam unicamente, a saber:

- Elemento gráfico, que o representa visualmente;
- Conjunto de atributos descritivos;
- Instante no tempo em que existe.

Considerando os três itens citados anteriormente, um conjunto de observações de dados espaciais (ou uma coleção de conjuntos) obrigatoriamente conterá contêineres relativos a cada um dos ditos itens. Em cada contêiner, todas as instâncias dos elementos observados devem ser armazenadas, e uma relação lógica entre os diversos contêineres deve ser estabelecida de modo a validar o descritor de metadados.

Todo conjunto de observações, logicamente, está armazenado em uma fonte de dados, ou, como já definido, em um recurso. Desta forma, um descritor do recurso deve, necessariamente, conter também informações acerca do local onde o mesmo é disponibilizado, de modo a permitir que clientes o encontrem. A Fig. 4.1 apresenta resumidamente a estrutura de um RDF de metadados espaço-temporais para melhor entendimento.

Cada item constituinte de uma observação é composto por outros elementos menores. O detalhamento dos mesmos é apresentado na sequência.

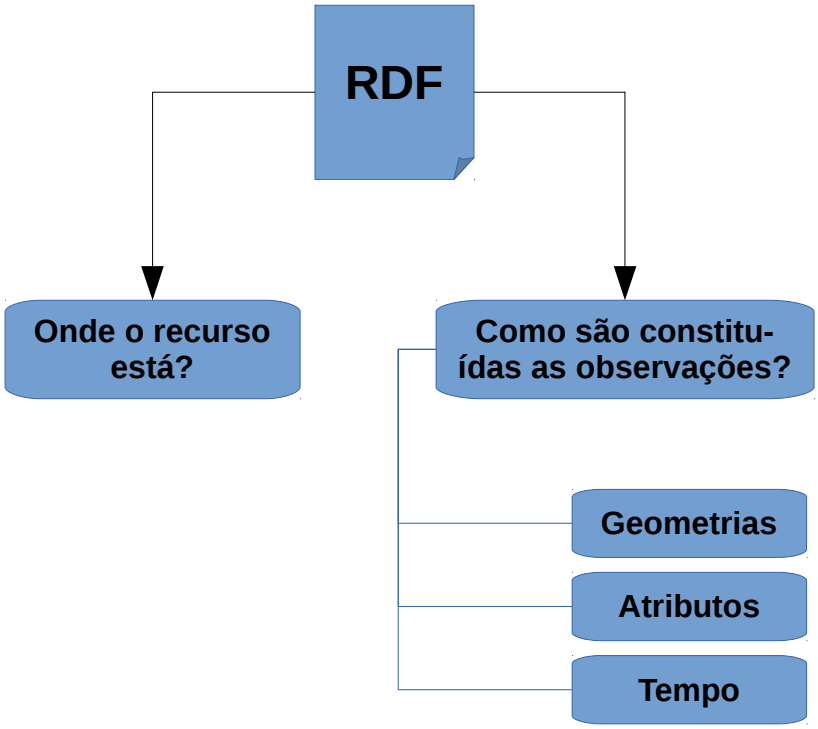


Figura 4.1 – Conjunto de informações armazenadas no descritor de metadados.

(Fonte: o autor)

## 4.2 Elementos gráficos da observação

O conjunto de representações dos dados espaço-temporais é armazenado no recurso segundo algum modelo específico, inerente ao tipo de recurso em questão. De todo modo, sendo este coerente com os padrões da OGC, possuirá determinados elementos constituintes, relevantes para estabelecer as relações temporais entre os dados desejada, a saber (as referências serão úteis a posteriori):

- (1a) Tipo do local onde as representações estão armazenadas no recurso.
- (1b) Nome do local onde as representações estão armazenadas no recurso.
- (1c) Nome do atributo, no local definido por 1c e 1d, que contém os identificadores únicos das representações.
- (1d) Retângulo envolvente do espaço geográfico representado.

Foi utilizada a denominação de representação por um motivo simples: o elemento gráfico em questão pode ser vetorial ou matricial.

Para fins de implementação, os seguintes nomes foram adotados:

- RepresentationInfo → contêiner de representações gráficas
- RepContainer → 1a
- RepName → 1b
- RepID → 1c
- BoundingBox → 1d

Visualmente, os itens elencados são apresentados na Fig. 3.2.

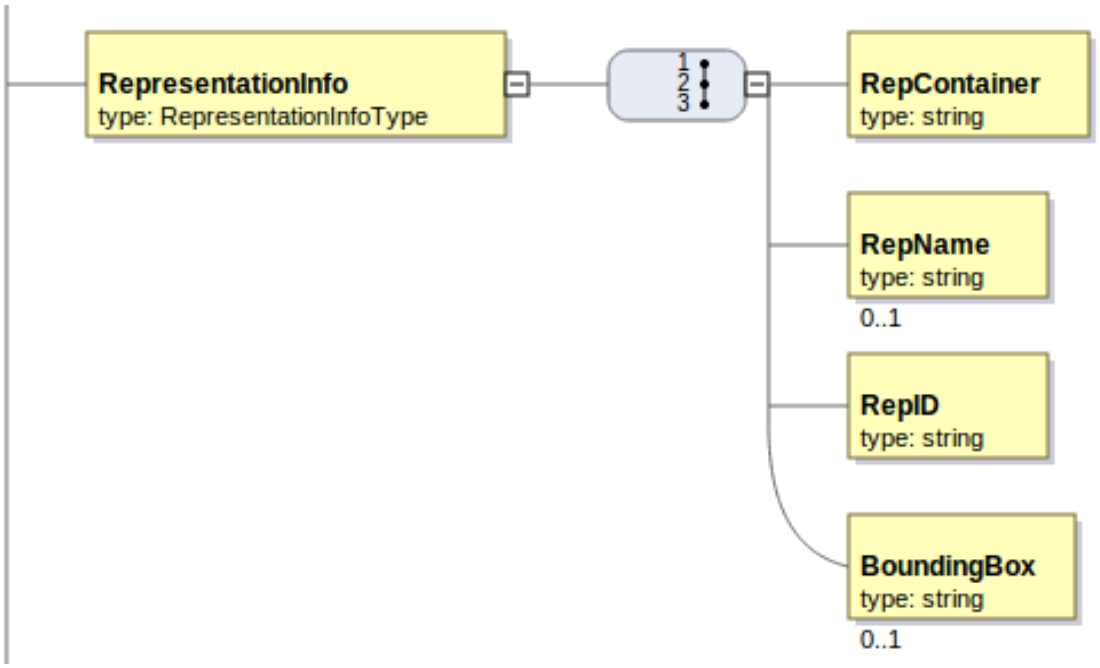


Figura 4.2 – Modelagem dos metadados das representações gráficas dos dados espaciais em arquivos RDF. (Fonte: o autor)

## 4.3 Atributos descritivos

Dados espaciais, por sua definição, possuem (ou podem possuir) um conjunto de atributos alfanuméricos que o descrevem. Considerando a possibilidade, os itens necessários à abordagem proposta são os que seguem:

- (2a) Tipo do local onde os atributos estão armazenados no recurso.
- (2b) Nome do local onde os atributos estão armazenados no recurso.
- (2c) Atributo que funciona como *link* entre os demais e a representação gráfica do dado.

Com base no tipo de recurso, alguns dos predicados acima podem ou não ser necessários, o que consequentemente elimina a necessidade de existência de triplas com os mesmos e sem qualquer objeto. Dois casos podem ser tomados como exemplo:

- Arquivos KML (*Keyhole Markup Language*) → basta 2a ser preenchido; os outros objetos são ignorados.
  - A *tag* do KML não possui um "tipo", conceitualmente falando. Possui apenas um nome, que pode ser encontrado via *parsing*.
- Banco de dados PostGIS → todos os objetos são necessários
  - 2a → Os atributos podem estar separados das representações, em outra tabela, ou juntos às mesmas, na mesma tabela.
    - Primeiro caso → O valor será "outratabela" (ou termo de mesmo significado).
    - Segundo caso → O valor será "mesmatabela" (ou termo de mesmo significado).
  - 2b → Nome da tabela que contém os atributos (item anterior, primeiro caso), ou nome da coluna a partir da qual todas serão consideradas atributos (item anterior, segundo caso).

A análise de necessidade de existência de um determinado predicado deverá ser processada pelo cliente, de modo mais profundo visto como o conjunto de bibliotecas computacionais que faz o acesso ao recurso.

De forma visual, os itens 2a, 2b e 2c são apresentados na Fig. 4.3.

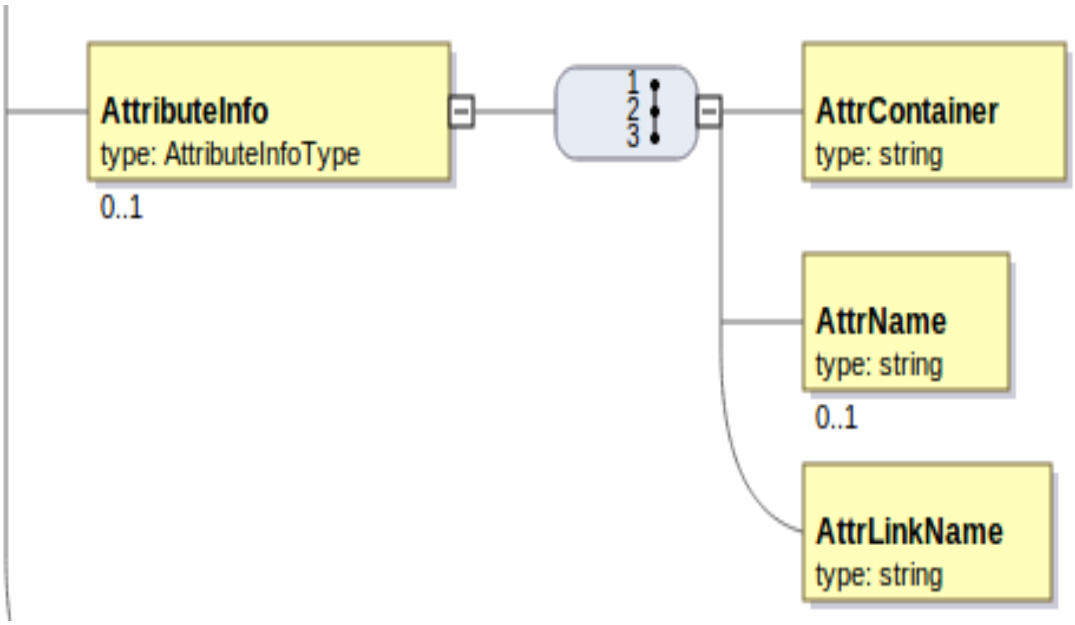


Figura 4.3 – Modelagem dos metadados dos atributos dos dados espaciais em arquivos RDF. (Fonte: o autor)

## 4.4 Informação temporal

O constituinte de maior importância de observações espaço-temporais é justamente o que a define como tal: a componente do tempo. Para descrevê-lo nos metadados, é preciso ter em mãos as seguintes informações:

- (3a) Tipo do local onde as informações de tempo estão armazenadas no recurso.
- (3b) Nome do local onde as informações de tempo estão armazenadas no recurso.
- (3c) Nome do atributo, em 3b, que contém a informação temporal.
- (3d) Elemento que liga a informação temporal com as demais – representação e atributos – quando não se encontra armazenada na mesma estrutura destas.
- (3e) Formato da informação do tempo.
- (3f) Resolução temporal do conjunto de observações considerado.
- (3g) Extensão temporal do conjunto de observações.

Do mesmo modo que ocorreu no tópico anterior, alguns dos predicados acima podem não ser necessários, conforme o tipo de recurso considerado. Alguns são opcionais para todos os tipos, pois podem ser estabelecidos por processamento, no cliente, dos dados recuperados. Assim, tomando-se os mesmos exemplos de armazenamento do tópico anterior, tem-se os seguintes casos:

- Arquivos KML → É suficiente apenas 3b.
- Banco de dados PostGIS
  - Mandatórios → 3a, 3b, 3c
  - Eventualmente mandatórios → 3d
    - Depende dos objetos de 3a e 3b, pois se forem iguais significa que a informação temporal pertence ao mesmo contêiner ou de representações ou de atributos.
  - Opcionais → 3e, 3f, 3g

Mais uma vez, a análise de opcionalidade e requisito é deixada para o sistema cliente.

A representação gráfica dos constituintes da informação temporal da observação é ilustrada na Fig. 4.4.

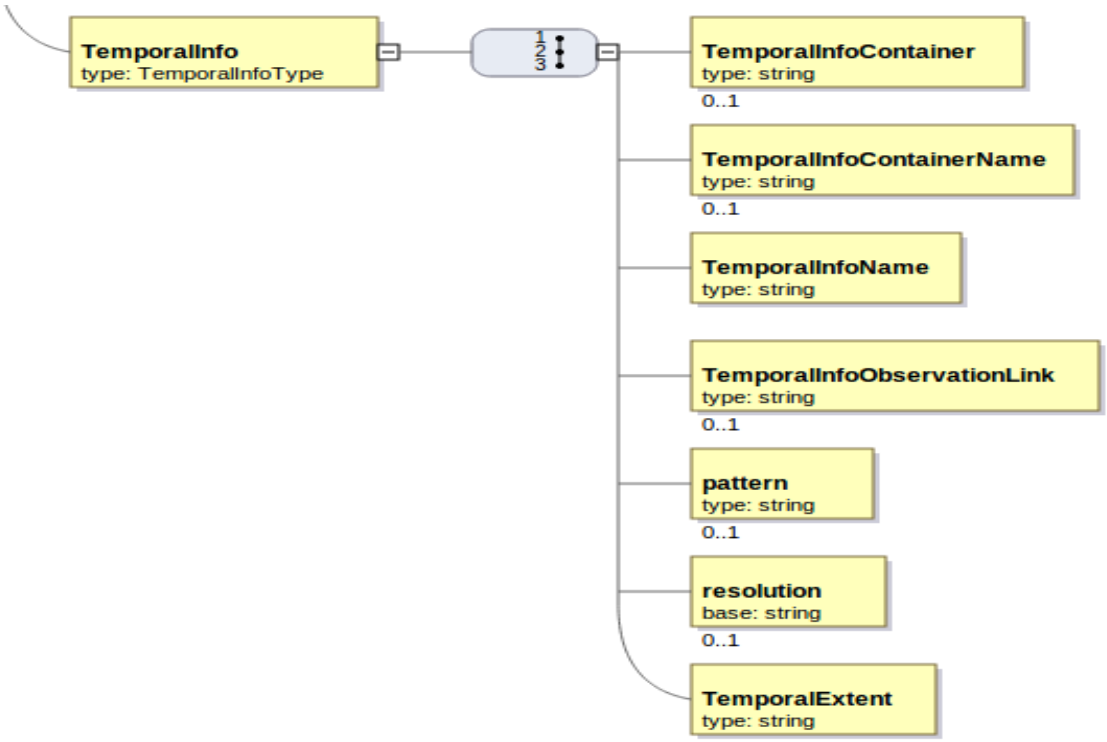


Figura 4.4 – Modelagem dos metadados das informações temporais das observações, para aplicação em arquivos RDF. (Fonte: o autor)

## 4.5 Informações sobre o recurso

Uma vez que a estruturação do recurso foi estabelecida, deixando claro de que forma os dados armazenados são correlatos no tempo, é preciso descrever as informações referentes ao recurso em si, tais como:

- (1a) Nome do recurso.
- (1b) Tipo do recurso (formato do arquivo, tipo do banco de dados, etc.).
- (1c) Parâmetros de acesso ao recurso.

As informações acima vinculam o arquivo RDF de metadados a um recurso específico, válido para sua estrutura de armazenamento e para os seus daods armazenados.

A Fig. 4.5 apresenta graficamente os constituintes da informação sobre o recurso.

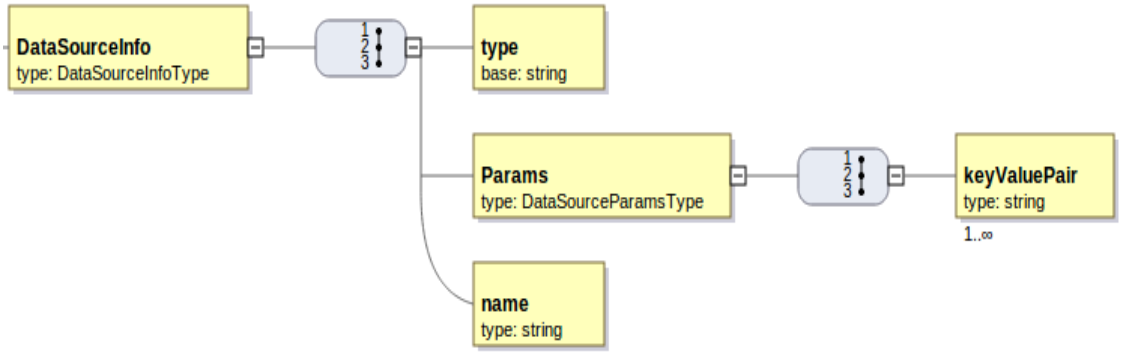


Figura 4.5 – Modelagem dos metadados das informações acerca do recurso, que vinculam o RDF a um recurso específico. (Fonte: o autor)

# V IMPLEMENTAÇÃO DA PROPOSTA EM XML E RDF/XML

## 5.1 Modelo completo da proposta

Com base na Fig. 5.1, e com referência às Fig. 4.2, 4.3, 4.4 e 4.5, obtém-se o modelo completo da proposta levantada conforme a Fig. 4.1.

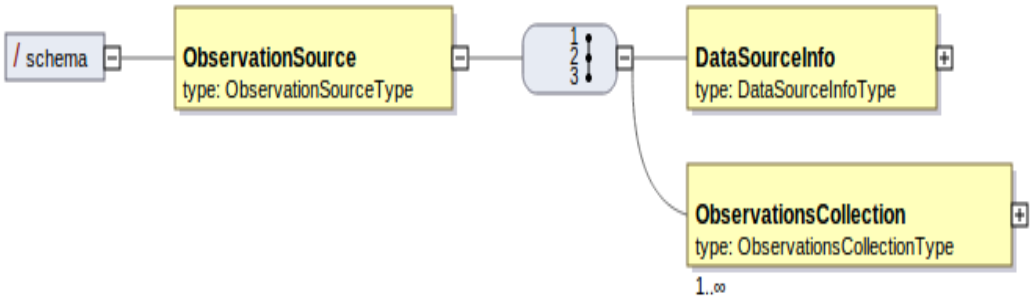


Figura 5.1 – Resumo do modelo completo da proposta da estrutura de metadados.

(Fonte: o autor)

As coleções de observação, por sua vez, são detalhadas na Fig. 5.2. Percebe-se que há um elemebto identificador (“CollectionIdentifier”), cuja função é guardar o nome do atributo que contém o identificador das representações, aquele que as liga com os demais constituintes da observação. Pode ser dispensável, dependendo do tipo de recurso.

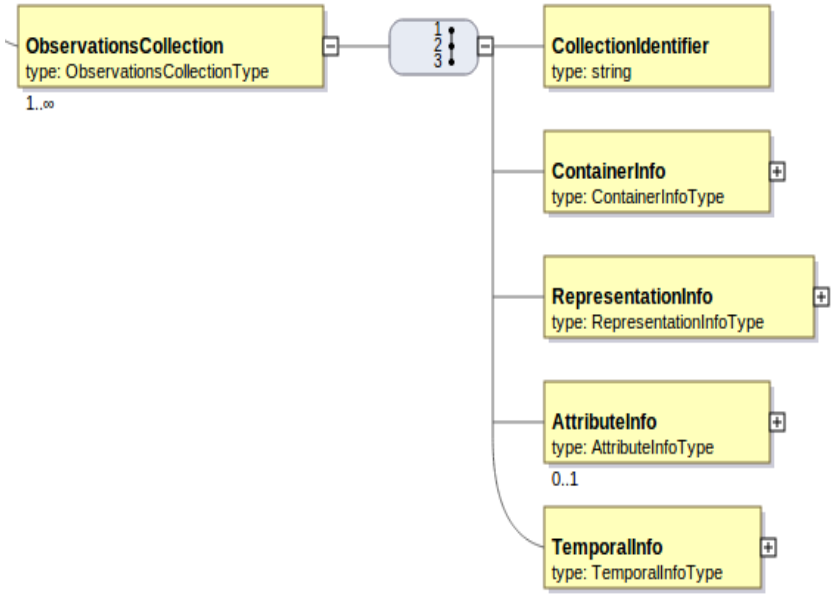


Figura 5.2 – Detalhamento da estrutura da coleção de observações.

(Fonte: o autor)

## 5.2 Implementação em XML

A proposta visa estabelecer um modelo para descrição de metadados. Assim, sua implementação gera um esquema XML. O código referente a este XSD é o constante do Anexo I.

## 5.3 Conversão do XSD para RDFs

Uma vez elaborado o XSD, procedeu-se à sua conversão para as particularidades da modelagem RDF. Cabe ressaltar que há o uso de *namespaces* atualmente inexistentes, uma vez que para o arquivo RDFs ser utilizado deve estar publicado em um servidor. Isto, contudo, não interfere nos resultados alcançados. O referido RDFs encontra-se no Anexo II.

## 5.4 Aplicativos utilizados

Para escrita e validação do XSD gerado no item anterior, utilizou-se o *software* EditiX 2008 Service Pack 2 Free Edition, da empresa JAPISoft SARL, em sua versão para sistema operacional Linux. Assim, utilizou-se um ambiente computacional totalmente livre.

# VI ANÁLISE DE ADEQUABILIDADE DA PROPOSTA

## 6.1 KML

Para avaliar se a proposta é aplicável a um arquivo KML com dados espaço-temporais, tomou-se como referência o mesmo arquivo de observações utilizado por Ferreira et al. (2012). Com base nele, a análise se deu pela tentativa de se relacionar cada elemento da proposta a um elemento inerente ao KML. O resultado desta tentativa é mostrado na Fig. 6.1.

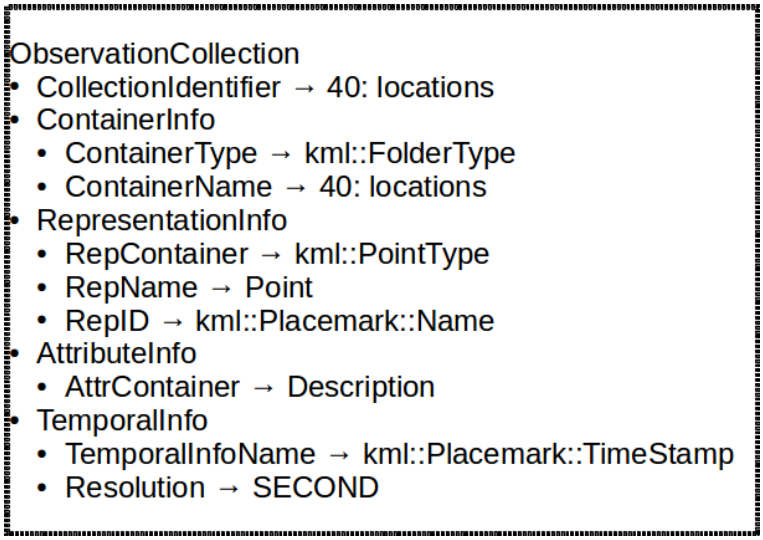


Figura 6.1 – Descrição dos metadados das coleções de observações.

(Fonte: o autor)

O mesmo procedimento para o contêiner de informações inerentes ao recurso em si é desnecessário, haja vista que os elementos do dito contêiner não são relacionados com elementos internos ao arquivo.

## 6.2 PostGIS

Para o teste com banco de dados PostGIS, definiu-se uma situação hipotética, a saber:

- As representações estão armazenadas em uma tabela somente para este fim, tabela esta que pode possuir informação temporal.
- Os atributos inerentes às representações estão armazenadas em uma tabela somente para este fim, tabela esta que pode possuir informação temporal.
- As informações temporais, por inferência dos itens anteriores, estão em uma das duas tabelas citadas (ou em ambas), e não em uma terceira.

Na situação criada anteriormente, duas particularidades podem acontecer:

1. As representações serem vetoriais;
2. As representações serem matriciais.

Assim, para cada particularidade, pode-se ter os conjuntos de metadados que se seguem (alguns nomes são tomados como exemplo genérico).

1. Representações vetoriais

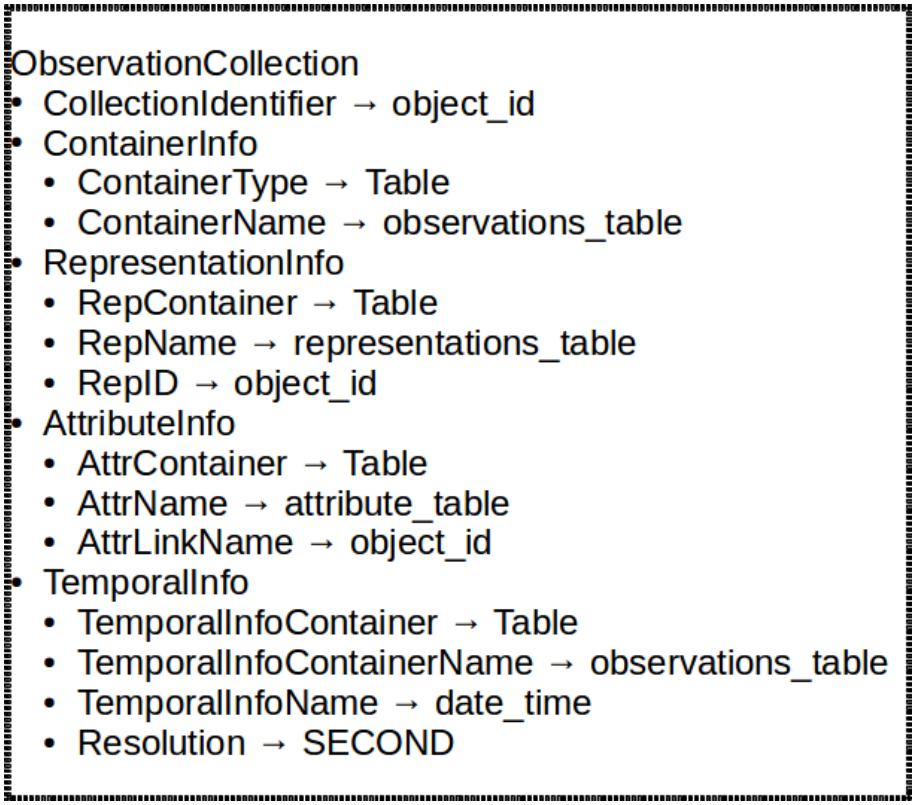


Figura 6.2 – Esquema de exemplo para metadados de recursos em banco de dados PostGIS, cujos dados espaço-temporais são representados por vetores. (Fonte: o autor)

Neste caso hipotético, cada integrante da coleção de observações está armazenado em uma estrutura diferente (Fig. 6.2). Pode acontecer (e não é incomum), contudo, a situação em que as geometrias estão armazenadas na mesma estrutura do contêiner de observações, juntamente com seus atributos e sua marcação temporal. Assim, conforme o Capítulo 4.3, ter-se-ia o conjunto de metatados descrito conforme a Fig. 6.3.

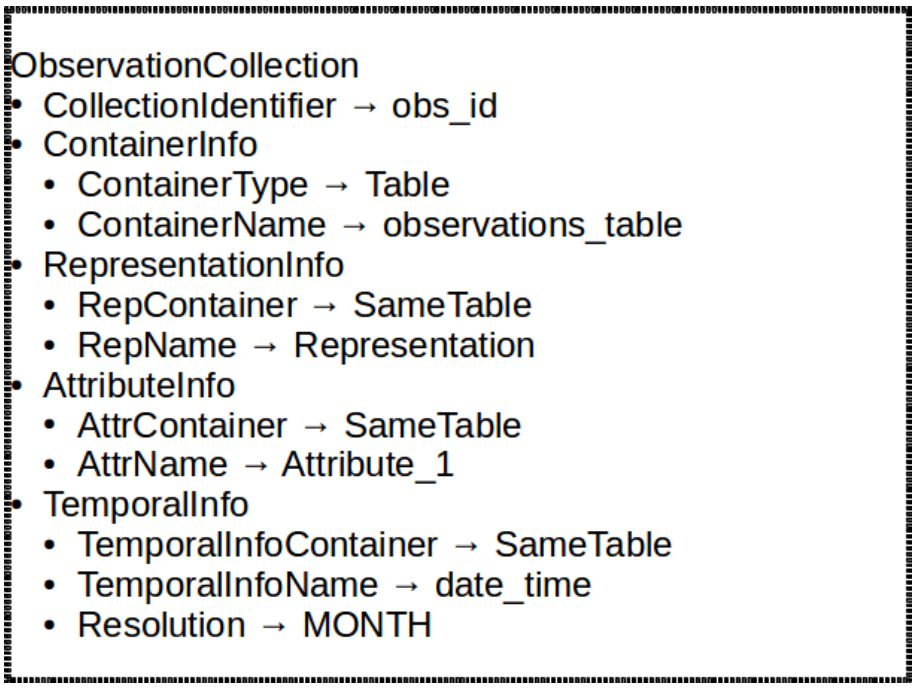


Figura 6.3 – Exemplo de estrutura de metadados de recursos em bancos de dados PostGIS no qual todos os elementos constituintes das observações são armazenados no mesmo contêiner (tabela). (Fonte: o autor)

2. Representações matriciais

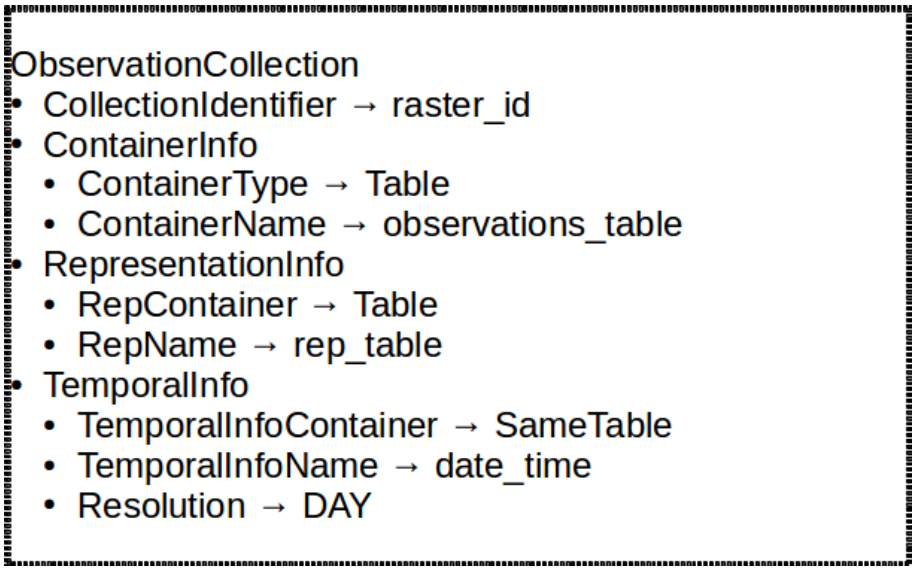


Figura 6.4 – Exemplo de estrutura de metadados de recursos em bancos de dados PostGIS, cujos dados espaço-temporais são matriciais. (Fonte: o autor)

No caso de dados matriciais representarem observações, a estrutura do arquivo de metadados pode ser estabelecida de acordo com o proposto na Fig. 6.4.

## 6.3 Derivação dos tipos de observação a partir do modelo proposto

Uma das finalidades de um modelo único de metadados é a de permitir que, a partir do mesmo, possa ser deduzido o tipo de observação recuperada. Os tipos possíveis, conforme Ferreira et al. (2013), são de trajetórias (*Trajectory*), coberturas (*Coverage*) – e séries de coberturas (*Coverage Series*) – e séries temporais (*Time Series*).

É possível, facilmente, verificar que a proposta deste trabalho atende ao requisito supracitado. Pela metodologia de interpretação de observações de Ferreira et al. (2013), adaptando-se ao modelo aqui proposto, constrói-se a tabela de interpretações, apresentada na Fig. 6.5.

| Dado fixo | Dado controlado | Dado medido | Resultado |
|---|---|---|---|
| RepresentationInfo | TemporalInfo | AttributeInfo | *Time Series*<br>*Coverage Series* |
| TemporalInfo | RepresentationInfo | AttributeInfo | *Coverage* |
| - | TemporalInfo | RepresentationInfo | *Trajectory* |

Figura 6.5 – Tabela de interpretações das observações, segundo a proposta elaborada.

(Fonte: o autor)

Pela figura acima, torna-se evidente que o modelo proposto atende aos requisitos mínimos para derivação dos diversos tipos de observações.

# VII CONCLUSÕES

A modelagem proposta para descrição dos metadados de recursos de observações espaço-temporais verificou-se correta, atendendo às necessidades impostas pela ausência de especificações técnicas padrões para tal. Contudo, por ser uma abordagem isolada (sem contribuições de terceiros), pode não ser suficiente para todos os produtores de dados. O importante, porém, é que contém uma estrutura mínima para os metadados, a partir da qual novos elementos possam ser incorporados, melhorando o modelo.

Outro aspecto a ressaltar é que a modelagem não foi testada de fato em um sistema. Não foram implementadas rotinas computacionais que interpretassem o RDF e recuperassem dados espaço-temporais, de modo a de fato empregar o modelo. Podem vir a surgir necessidades de adaptações do modelo, visando a correta implementação do seu interpretador.

De modo geral, a finalidade do trabalho foi atingida, e os resultados podem ser considerados satisfatórios.

# VIII TRABALHOS FUTUROS

Como foi apresentado nas conclusões, faltam algumas tarefas a serem executadas, de modo a colocar a modelagem para ser empregada. Além disto, outras avaliações de qualidade devem ser processadas, para incorporação ao modelo de requisitos não vislumbrados por este autor.

Assim, dentre os trabalhos a serem desenvolvidos futuramente, destacam-se os que seguem:

- Implementação de rotinas computacionais para interpretação do modelo RDF proposto.
- Construção de um “dicionário de termos espaço-temporais”, que defina de modo padronizado os sujeitos, predicados e objetos das triplas do RDF – necessário para garantir a interoperabilidade.
- Estudo da aplicabilidade do modelo a outros tipos de recursos diferentes dos abordados (KML e PostGIS).
- Comparação entre a proposta e o que existe (ou vier a existir) de especificação técnica ou discussão na OGC.

# IX REFERÊNCIAS BIBLIOGRÁFICAS


BRAY, Tim; COWAN, John; MALER, Eve; PAOLI, Jean; SPERBERG-MCQUEEN, C. M.; YERGEAU, François. *Extensible Markup Language (XML) 1.1*. World Wide Web Consortium Recommendation, 2006 [online]. Disponível em: <http://www.w3.org/TR/xml11>. Acessado em: 23 jun. 2014.

COX, S. *Observations and Measurements – XML Implementation (OGC 10–025r1)*. OGC Implementation Specification, 2011 [internet]. Disponível em: <http://www.opengis.net/doc/IS/OMXML/2.0>. Acessado em: 20 jun. 2014.

FERREIRA, Karine Reis; CAMARA, Gilberto; MONTEIRO, Antônio Miguel Vieira. *An algebra for spatiotemporal data: from observations to events*. **Transactions in GIS**, v. 18, n. 2, p. 253-269, 2013.

FERREIRA, Karine Reis; CAMARA, Gilberto; MONTEIRO, Antônio Miguel Vieira; VINHAS, Lubia. *Moving objects and spatial data sources*. **Revista Brasileira de Cartografia**, n. 64/4, 2012.

ISO – International Organization for Standardization. *ISO 19115-1:2014 – Geographic Information Metadata*. 2014 [internet]. Disponível em: <http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?snumber=53798>. Acessado em: 20 jun. 2014.

OGC – Open Geospatial Consortium. *OGC Standards*. 2012 [internet]. Disponível em: <http://www.opengeospatial.org/standards/is>. Acessado em: 20 jun. 2014.

PERRY, Matthew; JAIN, Prateek; SHETH, Amit P. *Sparql-st: Extending sparql to support spatiotemporal queries*. In: ***Geospatial semantics and the semantic web***. Springer US, 2011. p. 61-86.

WOOD, D.; ZAIDMAN, M.; RUTH, L.; HAUSENBLAS, M. *Linked Data:* Structured Data on the Web. Shelter Island-NY: Manning Publications Co., 2014. 336 p.

**ANEXO I**

**CÓDIGO XML DA MODELAGEM PROPOSTA**

```
<?xml version="1.0" encoding="UTF-8"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
	<xs:element name="ObservationSource" type="ObservationSourceType"/>
		<xs:complexType name="ObservationSourceType">
			<xs:sequence>
				<xs:element name="DataSourceInfo" type="DataSourceInfoType"/>
				<xs:element maxOccurs="unbounded"
name="ObservationsCollection" type="ObservationsCollectionType"/>
			</xs:sequence>
		</xs:complexType>

		<xs:complexType name="DataSourceInfoType">
			<xs:sequence>
				<xs:element name="type">
					<xs:simpleType>
						<xs:restriction base="xs:string">
							<xs:enumeration value="KML"/>
							<xs:enumeration value="POSTGIS"/>
						</xs:restriction>
					</xs:simpleType>
				</xs:element>
				<xs:element name="Params" type="DataSourceParamsType"/>
				<xs:element name="name" type="xs:string"/>
			</xs:sequence>
		</xs:complexType>

		<xs:complexType name="DataSourceParamsType">
			<xs:sequence>
				<xs:element maxOccurs="unbounded" name="keyValuePair"
type="xs:string"/>
			</xs:sequence>
		</xs:complexType>

		<xs:complexType name="ObservationsCollectionType">
			<xs:sequence>
				<xs:element name="CollectionIdentifier" type="xs:string"
minOccurs="1" maxOccurs="1"/>
				<xs:element name="ContainerInfo" type="ContainerInfoType"
minOccurs="1" maxOccurs="1"/>
				<xs:element name="RepresentationInfo"
type="RepresentationInfoType" minOccurs="1" maxOccurs="1"/>
				<xs:element name="AttributeInfo" type="AttributeInfoType"
minOccurs="0" maxOccurs="1"/>
				<xs:element name="TemporalInfo" type="TemporalInfoType"
maxOccurs="1"/>
```

```
		</xs:sequence>
	</xs:complexType>

	<xs:complexType name="ContainerInfoType">
		<xs:sequence>
			<xs:element name="ContainerType">
				<xs:simpleType>
					<xs:restriction base="xs:string">
						<xs:enumeration value="Table"/>
						<xs:enumeration value="FolderType"/>
						<xs:enumeration value="DocumentType"/>
					</xs:restriction>
				</xs:simpleType>
			</xs:element>
			<xs:element name="ContainerName" type="xs:string"/>
		</xs:sequence>
	</xs:complexType>

	<xs:complexType name="RepresentationInfoType">
		<xs:sequence>
			<xs:element name="RepContainer" type="xs:string"
minOccurs="1"/>
			<xs:element name="RepName" type="xs:string" minOccurs="0"/>
			<xs:element name="RepID" type="xs:string" minOccurs="1"/>
			<xs:element name="BoundingBox" type="xs:string"
minOccurs="0"/>
		</xs:sequence>
	</xs:complexType>

	<xs:complexType name="AttributeInfoType">
		<xs:sequence>
			<xs:element name="AttrContainer" type="xs:string"
minOccurs="1"/>
			<xs:element name="AttrName" type="xs:string" minOccurs="0"/>
			<xs:element name="AttrLinkName" type="xs:string"
minOccurs="1"/>
		</xs:sequence>
	</xs:complexType>

	<xs:complexType name="TemporalInfoType">
		<xs:sequence>
			<xs:element name="TemporalInfoContainer" type="xs:string"
minOccurs="0"/>
			<xs:element name="TemporalInfoContainerName" type="xs:string"
minOccurs="0"/>
			<xs:element name="TemporalInfoName" type="xs:string"/>
			<xs:element name="TemporalInfoObservationLink" type="xs:string"
minOccurs="0"/>

			<xs:element minOccurs="0" name="pattern" type="xs:string"/>
```

```
                        <xs:element minOccurs="0" name="resolution">
                            <xs:simpleType>
                                <xs:restriction base="xs:string">
                                    <xs:enumeration value="SECOND"/>
                                    <xs:enumeration value="MINUTE"/>
                                    <xs:enumeration value="HOUR"/>
                                    <xs:enumeration value="DAY"/>
                                    <xs:enumeration value="MONTH"/>
                                    <xs:enumeration value="YEAR"/>
                                </xs:restriction>
                            </xs:simpleType>
                        </xs:element>
                        <xs:element name="TemporalExtent" type="xs:string"/>
                    </xs:sequence>
                </xs:complexType>
</xs:schema>
```

**ANEXO II**

**CÓDIGO RDF/XML DA MODELAGEM PROPOSTA**

```
<?xml version="1.0" encoding="UTF-8"?>

<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/TR/WD-rdf-schema#" xml:base="http://www.terralib.org/rdf/st">

<!--  Defining classes and subclasses -->

        <rdfs:Class rdf:ID="ObservationSourceType">
        </rdfs:Class>

        <rdfs:Class rdf:ID="DataSourceInfoType">
                <rdfs:subClassOf rdf:resource="#ObservationSourceType"/>
        </rdfs:Class>

        <rdfs:Class rdf:ID="DataSourceParamsType">
                <rdfs:subClassOf rdf:resource="#DataSourceInfoType"/>
        </rdfs:Class>

        <rdfs:Class rdf:ID="ObservationsCollectionType">
                <rdfs:subClassOf rdf:resource="#ObservationSourceType"/>
        </rdfs:Class>

        <rdfs:Class rdf:ID="ContainerInfoType">
                <rdfs:subClassOf rdf:resource="#ObservationsCollectionType"/>
        </rdfs:Class>

        <rdfs:Class rdf:ID="RepresentationInfoType">
                <rdfs:subClassOf rdf:resource="#ObservationsCollectionType"/>
```

```
	</rdfs:Class>

	<rdfs:Class rdf:ID="AttributeInfoType">
		<rdfs:subClassOf rdf:resource="#ObservationsCollectionType"/>
	</rdfs:Class>

	<rdfs:Class rdf:ID="TemporalIInfoType">
		<rdfs:subClassOf rdf:resource="#ObservationsCollectionType"/>
	</rdfs:Class>


<!-- Defining properties of already defined classes -->

	<rdfs:Property rdf:ID="CollectionIdentifier">
		<rdfs:domain rdf:resource="#ObservationCollectionType" />
		<rdfs:range rdf:resource="rdf:literal" />
	</rdfs:Property>

	<rdfs:Property rdf:ID="DataSourceInfo">
		<rdfs:domain rdf:resource="#ObservationSourceType" />
		<rdfs:range rdf:resource="#DataSourceInfoType" />
	</rdfs:Property>

	<rdfs:Property rdf:ID="ObservationInfo">
		<rdfs:domain rdf:resource="#ObservationSourceType" />
		<rdfs:range rdf:resource="ObservationInfoType" />
	</rdfs:Property>

	<!-- There is the need to use OWL for restriction of accepted values, not yet defined. -->
	<rdfs:Property rdf:ID="type">
		<rdfs:domain rdf:resource="#DataSourceInfoType" />
		<rdfs:range rdf:resource="rdf:literal" />
	</rdfs:Property>

	<rdfs:Property rdf:ID="Params">
		<rdfs:domain rdf:resource="#DataSourceInfoType" />
		<rdfs:range rdf:resource="#DataSourceParamsType" />

	</rdfs:Property>

	<rdfs:Property rdf:ID="name">
	<!-- As various classes shares this property, the domain is not specified. Doing so, any class could have this property. -->
		<rdfs:range rdf:resource="rdf:literal" />
	</rdfs:Property>

	<rdfs:Property rdf:ID="keyValuePair">
		<rdfs:domain rdf:resource="#DataSourceParams" />
		<rdfs:range rdf:resource="rdf:literal" />
	</rdfs:Property>
```

```
<rdfs:Property rdf:ID="ContainerInfo">
        <rdfs:domain rdf:resource="#ObservationInfoType" />
        <rdfs:range rdf:resource="#ContainerInfoType" />
</rdfs:Property>

<rdfs:Property rdf:ID="RepresentationInfo">
        <rdfs:domain rdf:resource="#ObservationInfoType" />
        <rdfs:range rdf:resource="#RepresentationInfoType" />
</rdfs:Property>

<rdfs:Property rdf:ID="AttributeInfo">
        <rdfs:domain rdf:resource="#ObservationInfoType" />
        <rdfs:range rdf:resource="#AttributeInfoType" />
</rdfs:Property>

<rdfs:Property rdf:ID="TemporalInfo">
        <rdfs:domain rdf:resource="#ObservationInfoType" />
        <rdfs:range rdf:resource="#TemporalInfoType" />
</rdfs:Property>

<rdfs:Property rdf:ID="ContainerType">
        <rdfs:domain rdf:resource="#ContainerInfoInfoType" />
        <rdfs:range rdf:resource="rdf:literal" />
</rdfs:Property>

<rdfs:Property rdf:ID="ContainerName">
        <rdfs:domain rdf:resource="#ContainerInfoType" />
        <rdfs:range rdf:resource="rdf:literal" />
</rdfs:Property>

<rdfs:Property rdf:ID="RepContainer">
        <rdfs:domain rdf:resource="#RepresentationInfoType" />
        <rdfs:range rdf:resource="rdf:literal" />
</rdfs:Property>

<rdfs:Property rdf:ID="RepName">
        <rdfs:domain rdf:resource="#RepresentationInfoType" />
        <rdfs:range rdf:resource="rdf:literal" />
</rdfs:Property>

<rdfs:Property rdf:ID="RepID">
        <rdfs:domain rdf:resource="#RepresentationInfoType" />
        <rdfs:range rdf:resource="rdf:literal" />
</rdfs:Property>

<rdfs:Property rdf:ID="BoundingBox">
        <rdfs:domain rdf:resource="#RepresentationInfoType" />
        <rdfs:range rdf:resource="rdf:literal" />
</rdfs:Property>
<rdfs:Property rdf:ID="AttrContainer">
```

```
            <rdfs:domain rdf:resource="#AttributeInfoType" />
            <rdfs:range rdf:resource="rdf:literal" />
        </rdfs:Property>

        <rdfs:Property rdf:ID="AttrName">
            <rdfs:domain rdf:resource="#AttributeInfoType" />
            <rdfs:range rdf:resource="rdf:literal" />
        </rdfs:Property>

        <rdfs:Property rdf:ID="AttrLinkName">
            <rdfs:domain rdf:resource="#AttributeInfoType" />
            <rdfs:range rdf:resource="rdf:literal" />
        </rdfs:Property>

        <rdfs:Property rdf:ID="TemporalInfoContainer">
            <rdfs:domain rdf:resource="#TemporalInfoType" />
            <rdfs:range rdf:resource="rdf:literal" />
        </rdfs:Property>

        <rdfs:Property rdf:ID="TemporalInfoContainerName">
            <rdfs:domain rdf:resource="#TemporalInfoType" />
            <rdfs:range rdf:resource="rdf:literal" />
        </rdfs:Property>

        <rdfs:Property rdf:ID="TemporalInfoName">
            <rdfs:domain rdf:resource="#TemporalInfoType" />
            <rdfs:range rdf:resource="rdf:literal" />
        </rdfs:Property>

        <rdfs:Property rdf:ID="TemporalInfoObservationLink">
            <rdfs:domain rdf:resource="#TemporalInfoType" />
            <rdfs:range rdf:resource="rdf:literal" />
        </rdfs:Property>

        <rdfs:Property rdf:ID="pattern">
            <rdfs:domain rdf:resource="#TemporalInfoType" />
            <rdfs:range rdf:resource="rdf:literal" />
        </rdfs:Property>

        <!-- There is the need to use OWL for restriction of accepted values -->
        <rdfs:Property rdf:ID="resolution">
            <rdfs:domain rdf:resource="#TemporalInfoType" />
            <rdfs:range rdf:resource="xsd:string" />
        </rdfs:Property>

        <rdfs:Property rdf:ID="TemporalExtent">
            <rdfs:domain rdf:resource="#TemporalInfoType" />
            <rdfs:range rdf:resource="rdf:literal" />
        </rdfs:Property>
</rdf:RDF>
```