A Mechanism for Propagation of Semantic Annotations of Multimedia Content

Gilberto Zonta Pastorello Jr, Jaudete Daltio, Claudia Bauzer Medeiros
Institute of Computing, University of Campinas, 13084-971. Campinas, Brazil.
gilbertozp@acm.org, jaudete@gmail.com, cmbm@ic.unicamp.br

***Abstract*— Scientific research is producing and consuming large volumes of multimedia data at an ever growing rate. Data annotations are used, among others, to provide context information and enhance content management, making it easier to interpret and share data. However, raw multimedia data often needs to go through complex processing steps before it can be consumed. During these transformation processes, original annotations from the production phase are often discarded or ignored, since their usefulness is usually limited to the first transformation step. New annotations must be made at each step, and associated with the final product, a time consuming task often carried out manually. The task of systematically associating new annotations to the result of each data transformation step is known as *annotation propagation*. This paper introduces techniques for structuring and propagating annotations, in parallel to the data transformation processes, thereby alleviating the overhead and decreasing the errors introduced by manual annotation. This helps the construction of new annotated multimedia data sets, preserving contextual information. The solution is based on: (i) the notion of semantic annotations; (ii) a set of transformations rules, based on ontological relations; and, (iii) workflows that deal with interrelated processing steps.**

***Index Terms*— Annotation propagation, semantic annotations, metadata evolution, ontological relations.**

I. Introduction

Scientific applications are producing and consuming ever growing volumes of multimedia data, which may vary from data generated by sensors (e.g., aboard satellites or ground-based sensors) to video and sound recordings. In this scenario, scientists constantly need to share and reuse their data sets, being hampered by the wide spectrum of data production devices, actors and contexts that are involved in a lifecycle that *produces*, *transforms* and *consumes* data.

In order to decrease this heterogeneity scenario, scientists frequently adopt metadata and annotations as the primary means of describing data sets – e.g., [1]. Metadata are, often, text fields associated to data (e.g., in a file header) to be directly applied in automated data management tasks, such as indexing, searching, or context integration [2–4]. Annotations, on the other hand, are more flexible, often representing personal remarks created by data producers and/or consumers [5, 6]. However, annotations are also more limited when considering management tasks.

Albeit helpful in improving data interpretation, metadata/annotations become less useful, or even useless, as soon as a data set is transformed through some sort of processing function: the resulting data set requires new metadata/annotations. Roughly speaking, this characterizes the scenario for *metadata evolution* or *annotation propagation* [7, 8]. This poses the following problems: (1) how to propagate relevant metadata/annotations that would otherwise be discarded during a transformation? and, (2) how to support automatic creation of metadata/annotations for the transformed data, taking context into account? Our work contributes towards solving these two questions.

The traditional life-cycle for data sets is *(a) production – (b) transformation – (c) consumption*, where stage (b) may involve several steps. Most data interpretation tasks occur in the last stage. Adding metadata/annotations to the process improves the interpretation, and the cycle becomes *(a) production – (a') annotation – (b) transformation – (b') (re-)annotation – (c) consumption.* The main interest in this paper is on how to (partially or totally) automate stage (b'), thereby alleviating the overhead and decreasing the errors introduced by manual annotation. In particular, we are concerned with combining the notions of metadata, annotations and ontologies producing what we call *semantic annotations*, in which annotations are structured and defined in terms of references to ontology concepts and/or relationships. Terms from ontologies help provide contextual information.

Our approach starts by examining semantic annotations at stage (a') mentioned in the previous paragraph. Here, we assume that these annotations are available not only for data, but also for the operations that transform the data. We then propose a mechanism through which annotations are generated and associated with data sets produced by a data transformation operation - i.e., stage (b'). These new annotations are derived from the annotations made on the input data and on the operations, thanks to a set of propagation rules that are based on ontology terms and relationships.

A preliminary report on this work was published in [9]. This paper extends [9] in three aspects. First, it adds three propagation approaches to the one introduced in [9]. Second, while the data transformation operations in [9] are limited to one single input and one single output, here we consider operations with multiple inputs and outputs. Finally, we also analyze the chaining of such operations using scientific workflows. These extensions

enable our mechanism to deal with more complex data transformation operations as well as entire processes.

In more detail, we consider the operations that transform data to be workflow activities, and complex transformations are achieved by composing these activities into a workflow. Our propagation rules are applied in the sequence determined by the workflow: as data evolves as defined by the workflow, so will semantic annotations.

The main contributions of this paper are therefore: (i) a general definition for the annotation propagation problem, applicable to several different data transformation environments; (ii) an extensible ontology-guided technique for handling the annotation propagation problem using semantic annotations; and, (iii) a solution based on scientific workflows for dealing with complex transformation processes, including multiple input/output operations. As a consequence, discovery, sharing and reuse of multimedia data becomes easier. Though placed in the multimedia data management context, our solution can be extended to any environment where digital content is acquired, transformed and shared.

The remainder of the paper is organized as follows. Section II shows an example that we use to illustrate our proposal. Section III defines the annotation propagation problem. Section IV presents our solution to the problem using semantic annotations. Section V shows our proposal for dealing with multiple input/output operations and processes with several operations. Section VI revisits our running example, showing the application of our mechanism. Section VII discusses related work. Section VIII presents conclusions and future work.

II. Motivating Example

This section presents a motivating example that will be used throughout the text to help illustrate our proposal.

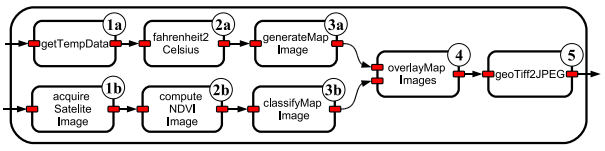

Figure 1. Data transformation process involving multimedia content.

Figure 1 illustrates a multimedia data transformation process in environmental modeling. This process combines satellite images with temperature readings (from ground sensors) for a given region and period, and generates JPEG images that show how temperature influences vegetation growth in the region. From a high level point of view, the steps are the following:
(1a) acquire temperature data (data streams) and (1b) satellite images for a given region (in a format called GeoTIFF[1]);
(2a) convert the temperature readings and (2b) compute the "greenness" of vegetation (using the so-called NDVI method[2]) on the satellite image generating a NDVI image;
(3a) generate a temperature map interpolating readings (e.g., using Thiessen polygons) and (3b) classify the regions in the NDVI image according to the greenness range – both maps generated at step 3 are GeoTIFF images;
(4) combine the two images into one; and,
(5) convert the resulting map into a JPEG image.

The classifyMapImage operation will be frequently used throughout the paper to highlight some particularities of our solution. Image classification is a very common procedure in scientific applications - e.g., medicine, environmental research, chemistry, astronomy, biodiversity. It applies image processing techniques to identify, within an image, clusters of neighboring pixels that belong to the same "class" – i.e., obey a given set of constraints, such as texture values or color intensity. The result of image classification is a new image where regions of similar pixels are singled out, usually by mapping all such pixels to a single value. In medicine, for instance, classification is used to identify tumors in an X-ray. In environmental research, the classification of a satellite image produces a new image with several clearly identifiable polygons (e.g., distinct colors), where each polygon stands for some sort of environmental entity – such as distinct types of vegetation. Classification is an example of an (often) lossy multimedia data transformation operation, in which output annotations are essential in ensuring data usability.

The image that results from running this process illustrates the correlation between temperature and vegetation conditions in the area and is sufficient for a high level view of this problem. If the process is run periodically, at the end we will have a set of images, portraying how such a correlation varies through time (e.g., seasonal changes). Input data sets (satellite image and temperature readings) are always annotated by the organization that produces them, using some consensual standard. However, at the end of the process, the output JPEG images will have no associated annotations, being thus unsuitable for any kind of scientific study on environmental conditions.

First, each output JPEG image should be annotated indicating that it is the result of combining satellite and sensor data. This may still not be enough – sensor type and calibration, satellite type and spectral band used must be informed. This contextual information is lost during the transformation process, unless all multimedia data involved in the process are manually annotated. This is difficult for large multi-step processes and impossible if parts of the process are controlled by different people or organizations - as is often the case when multimedia data are handled in scientific applications.

The more complex the data and the transformations performed, the greater the need for contextual information, and thus for detailed annotations.

[1] An image format where each pixel corresponds to a given location via its geographical coordinates.

[2] Normalized Difference Vegetation Index (NDVI) indicates the levels of live green vegetation, being usually computed from satellite images.

III. The Annotation Propagation Problem

A. Semantic Annotations

We combine characteristics of metadata and annotations into *semantic annotations*: using the structure from the first, filling its contents with references to ontologies, which provide the flexibility of the latter. Based on *Resource Description Framework* (RDF) structuring, we define semantic annotations as follows.

Annotation Units. An *annotation unit* a is a triple <s,p,o>, where s represents the subject being described, p represents a property of s, and, o represents a describing object or value.

Semantic Annotation. A *semantic annotation* M is a set of one or more annotation units, with at least one unit having as its subject the entity being described.

A semantic annotation is materialized as an RDF graph, which is represented as a set of RDF triples (*subject – predicate – object*); subject and predicate are identified by an URI[3] while the object may be an URI or a literal. Note that an object on one annotation unit may be itself a subject on another unit. This is the basic structuring element for semantic annotations. For space saving we omit the namespaces for terms in the text and figures.

We assume that a data transformation operation is a black-box that can be invoked; when provided appropriate input data, it produces output data. Semantic annotations are basically used to describe two entities in our solution: the data sets used as inputs and outputs and the interfaces of transformation operations. Figure 2 shows a transformation operation. The input data set D is annotated with M and the output data set D' is annotated with M'. The operation has its input interface I described by semantic annotation M_i and its output interface O described by M_o.

B. Annotation Propagation

Let us first consider the general annotation propagation problem. Let (T, I, O, D, D') denote an application of a data transformation operation T which has an input interface I and an output interface O, and is applied on a data set D, resulting in (derived) data D'. The definition for T was adapted from [10]. Also, let (τ, M_i, M_o, M, M') denote an application of an annotation transformation τ that manipulates M_i (the annotation of I), M_o (the annotation of O) and M (the annotation of D) to achieve M' (the derived annotation of D'). The annotation propagation problem is defined as follows.

The Annotation Propagation Problem. Consider a transformation T, with an input interface I and an output interface O, applied to a data set D, transforming it into another data set D'. Which transformation τ can generate the new annotations M', given the previous annotation M on the data, the annotation of the input interface M_i and the annotation of the output interface M_o?

[3]Or, to be more precise, a URIref, which is a URI that may have a fragment identifier (the symbol "#") at the end, for referencing parts of the URI.

If we now extend this definition to consider semantic annotations, the problem becomes: how to combine the sets of annotation units from the semantic annotation of the data set, the input interface and the output interface, to generate a new set of annotation units that will constitute the new semantic annotation. The mechanism to do that should ensure the consistency of the new set, as well as its completeness regarding the available annotations.

For the remainder of the text the term annotation refers to semantic annotation, unless otherwise specified.

As the operations considered are black-box operations, the annotation propagation in each step must be carried out outside the scope of the operation, i.e., by an external application. Thus, data transformation and annotation propagation do not interfere with each other.

Let us go back to our running example, and single out the classifyMapImage transformation operation that classifies an NDVI image generating as result a classified image. Figure 2 shows the association of annotations: M to the input data set (D), M' to the output data set (D'), M_i to the input interface of the operation (I) and M_o to the output interface of the operation (O). The bottom of Figure 2 portrays the transformation. The input data set (D) is an NDVI GeoTIFF image. The operation's input interface (I) takes one parameter (p1), and its output interface (O) produces one parameter (p2). The output data set (D') is the classified GeoTIFF image. These entities (data sets and interfaces) are described with semantic annotations, e.g., the pair (O, M_o) denotes that the semantic annotation M_o is associated with output interface O, similarly to (D, M), (I, M_i) and (D', M').

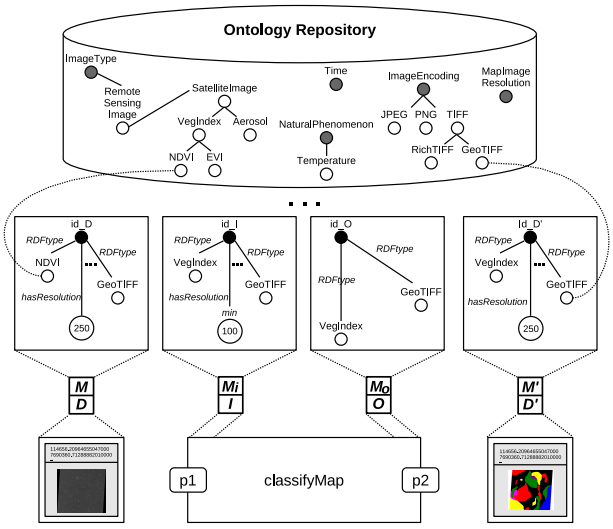

Figure 2. Data and interface annotations - a map (image obtained by processing satellite data) has its regions classified producing another multimedia dataset.

The top of the figure illustrates an ontology repository [11], a data space containing domain ontologies with the available contextual parameters and their relationships. The graphs in the boxes in the middle of the figure show how annotations are structured. M (at the left) is the annotation for the data set D: it is a graph

	M	M_i	M_o	derived content of M' with transformation: CMM_o	CMM_iM_o	OMM_o	OMM_iM_o
Encoding	*bitmap*	*bitmap*	*bitmap*	*bitmap*	*bitmap*	*bitmap*	*bitmap*
Image Type	*NDVI*	*V.I.*	*Class. V.I.*	*Class. V.I.*	*Class. V.I.*	*Class. V.I.*	*Class. V.I.*
Resolution (m)	*250*	*100+*	—	—	*250*	*250*	*250*
Timestamp	*20010323*	—	—	—	—	*20010323*	*20010323*
Compression	—	*LZW*	—	—	—	—	*LZW*

Figure 3. Values of M' for each type of propagation strategy – classifyMapImage.

rooted at id_D (the ID of the entity being described) and each edge defines the scope of one annotation *unit*. Units are stored as RDF triples. Thus, the annotation for D is {(id_D, RDFType, NDVI), (id_D, hasEncoding, GeoTIFF-bitmap), (id_D, capturedBy, Terra-MODIS) (id_D, usedBand, band4), (id_D, usedBand, band5), (id_D, hasOrbit/Point, 220/075F), (id_D, hasDatum, WGS84), (id_D, hasProjection, UTM_ELLIPSOID), (id_D, capturedOn, 20010323), ...}, indicating that it is an *NDVI image*, encoded in a *bitmap GeoTIFF*, captured by the *Terra-MODIS* satellite sensor, used *spectral bands 4* and *5*, and so on. By the same token, M_i says that I takes a parameter that should be a *Vegetation Index (VI) image*, encoded in a *bitmap GeoTIFF*, and so on; M_o indicates that O has as its output a *VI image*, encoded in a *bitmap GeoTiff*, and so on. Our mechanism is based in providing a consistent combination of these annotations (M, M_i and M_o), which generates the resulting propagated annotation (M').

C. Classification of Propagation Strategies

Before proceeding to the propagation mechanism, we introduce a classification of annotation propagation strategies. Figure 2 depicts the basic scenario (one transformation operation, one single data input and one single data output). M has elements with the values: "bitmap", "NDVI", "250", and "20010323", referencing, respectively, the ontology defined concepts *Encoding*, *Image Type*, *Resolution*, and *Timestamp*. Interface I expects a "Vegetation Index (VI) image", encoded as a "bitmap", with a resolution of at least "100" metres and can, as an optional feature, receive a GeoTIFF image compressed with the LZW algorithm. Interface O generates a "Classified VI image" encoded as a "bitmap". The data set D is a "NDVI image" which is converted into D', a "Classified VI image".

The question to be posed in a propagation scenario is: "which values should M' have?" Given (T, I, O, D, D') and (τ, M_i, M_o, M, M'), the code implementing T does not influence propagation, which should only consider relationships among M_i, M_o, M to produce M'.

We define the relationship between the input (I) and output (O) of a transformation operation to be determined by the ontological relationships among their annotations (M_i and M_o, respectively). Two factors can be used to classify the annotation propagation mechanism: (i) taking into account or not the relationship between the annotations on the input and the output of an operation; and (ii) taking into account or not annotations that cannot be matched against another (e.g., annotation units from M that cannot be compared to any other unit from M_o). Hence, four types of transformations are possible.

Regarding item (i), if the relationship between the input and output annotations is ignored, annotation propagation is based on directly applying the propagation rules considering only M and M_o. This is the first type of propagation: Closed $M - M_o$ (CMM_o). If, instead, the propagation considers the relationship, first we determine which terms from M_i influence M_o, and the rules are applied only to these terms. This is the second type of propagation: Closed $M - M_o$ through M_i (CMM_iM_o).

Considering (ii), it is possible that parts of one annotation (e.g., M) cannot be compared to any other part on the other annotation (e.g., M_o). Propagating any of these parts may produce richer annotations, at the cost of the possibility of introducing meaningless and/or erroneous annotations. Adding this degree of freedom to the previous two transformations, we have: the Open M–M_o propagation (called OMM_o) and the Open M–M_o through M_i propagation (called OMM_iM_o). Figure 3 exemplifies the contents of each kind of propagation for the example using the classifyMapImage operation. The elements on the table cells are objects for the annotation units.

For CMM_o the only comparable concepts (between M and M_o) were *Encoding* and *ImageType* – thus M' in this case contains "bitmap" and "Classified VI image". Propagation type CMM_iM_o had the *Resolution* concept matched between M_i and M_o, thus being included in the result. OMM_o propagated the M and M_o unmatched *Timestamp* concept, while for OMM_iM_o the *Timestamp* concept was first matched (between M_i and M_o) and then propagated. Actually, differentiation of behaviour between CMM_o and CMM_iM_o occurs when the input annotation is not directly reflected into the output annotation. Similarly, OMM_o differs from OMM_iM_o when one part of the input annotation does not match anything from the data annotations, as was the case in Figure 3: the "LZW" was attached to M' in the OMM_iM_o propagation strategy. This last situation only occurs with optional fields on an input annotation. Using either of the two open propagation strategies may lead to error or inconsistencies – as is the annotation "LZW" for the OMM_iM_o strategy, since the output of the classifyMapImage is not annotated as compressed. Error related issues are left for future work.

Section IV-A presents our mechanism for propagation strategies CMM_o and CMM_iM_o. Section IV-B discusses the propagation strategies OMM_o and OMM_iM_o.

IV. Semantic Annotation Propagation

This section presents our solution for the annotation propagation problem. In this solution, any data transformation operation performed on a data set must be accompanied by transformations on the associated annotations. No assumptions are made about the format or granularity of the data. The annotation propagation mechanism works equally for any kind of multimedia data. The section considers a basic data transformation process: a single operation with one data piece as input and one data piece as output. Section V then generalizes this proposal to the complex transformation operations with multiple inputs and outputs, and propagation through a chain of transformations.

Figure 4 schematically shows our solution for $\mathsf{CMM_o}$. The other approaches follow a similar schema, with the detail of applying the *Comparison/Generation* step first to $\mathsf{M_i}$ and $\mathsf{M_o}$, and then to the result and M for the cases of $\mathsf{CMM_iM_o}$ and $\mathsf{OMM_iM_o}$. At the first (leftmost) part are the annotations to be considered. The *Deconstruction* step breaks down the annotations into groups of annotation units - the <s, p, o> individual boxes in the second column of the figure. The second step, *Comparison/Generation*, makes a pairwise matching among the annotation units, generating the new annotation units in the third part of the figure. This matching must be performed between groups, e.g., in the figure units from M are matched with adequate units from $\mathsf{M_o}$. As will be seen, the comparisons are based on a selected set of ontological relations. The last step, *Reconstruction* combines the generated annotation units into a new semantic annotation (M'). Details of these steps are discussed next.

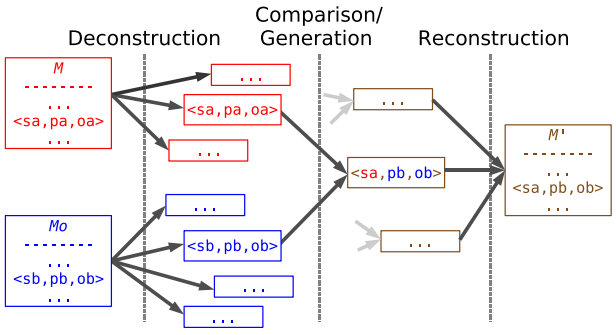

Figure 4. Schema for the proposed solution.

A. Closed Propagation Rules

Consider a transformation (T, I, O, D, D') with interface annotations $\mathsf{M_i}$ and $\mathsf{M_o}$ and let (D,M) and (D',M') be, respectively, its input and output data and annotations. The propagation problem, following the $\mathsf{CMM_o}$ strategy, can be simplified into the following issue: which mappings can be done between M and $\mathsf{M_o}$, meaning what were the effects of applying T to D with respect to its annotations?

We divided our solution this problem in two parts: (i) a set of abstract propagation rules to select pairs of annotations units for comparison; and, (ii) a set of ontological relations, each specifying how to compare a pair of annotation units and which annotation should be derived from the comparison. Our solution to the first part is a set of annotation propagation rules, presented next. Section IV-C shows to the second part. In this work, the annotation units are defined as RDF triples. However, an annotation unit could also be defined as a more complex structure, such as sub-paths, sub-trees or sub-graphs of the RDF graphs. In such cases, the propagation mechanism would have to be adapted to cope with these different annotation units. Exploring richer definitions of annotation units is left as future work.

Figure 5 shows a high level view of our propagation algorithm. Its input includes the annotations M and $\mathsf{M_o}$, and a set ($\Re$) of ontological relations ($\mathcal{R}_k$), such as the ones presented in Section IV-C. Its output is the resulting propagated semantic annotation (Δ).

```
1.  Δ ← ∅
2.  foreach R_k in ℜ do
3.      P_k ← ∅
4.      Ω ← F_k(M)
5.      Θ ← F_k(M_o)
6.      foreach a in Ω do
7.          foreach b in Θ do
8.              P_k ← P_k ∪ R_k(a, b)
9.      Δ ← Δ ∪ F_k^-1(P_k)
10. return Δ
```

Figure 5. Annotation Propagation Algorithm

Before the propagation rules can be applied, it is necessary to retrieve the annotation units from the RDF graph. Let us define a deconstruction function $\mathcal{F}$ to accomplish this. The result is a set of comparable units, which is represented by Ω (for M) and Θ (for $\mathsf{M_o}$), in lines 4 and 5, respectively. In this work this function simply singles out the RDF triples from the RDF graph, which are our basic comparison unit. Conversely, we also define a reconstruction function $\mathcal{F}^{-1}$ to recreate the RDF graph from the resulting unit(s) of the comparison of a pair of annotation units. In the algorithm, this is done in line 9, adding to the resulting semantic annotation Δ. The deconstruction and reconstruction steps are carried out for each ontological relation, since each relation may be based on different comparison units. In this paper, all ontological relations are based on annotation units as defined in Section III-A. For more complex units, $\mathcal{F}$ and $\mathcal{F}^{-1}$ would also increase in complexity.

The propagation rules simply enforce a systematic selection of a pair of annotation units and their comparison under a given ontological relation – lines 6-8 of the algorithm. These steps are repeated for all the selected ontological relations (i.e., all $\mathcal{R}_k$ in $\Re$). The results for all possible pairs are then returned (variable $\mathcal{P}_k$ on the algorithm). If the annotation units do not match under the ontological relation (line 8), the result of $\mathcal{R}_k(a, b)$ is empty. Intuitively, the rules recursively consider single

comparable annotation units, checking the compatibility between all pairs thereof and generating new units for the resulting set.

Therefore, to use the our propagation mechanism one must specify: (i) a set ($\Re$) of ontological relations ($\mathcal{R}_k$), each specifying the outcome of the comparison of two annotation units under this relation; (ii) a set of functions $\mathcal{F}_k$ to translate M into Ω and $\mathsf{M_o}$ into Θ; (iii) an inverse $\mathcal{F}_k^{-1}$ to translate $\mathcal{P}_k$ into Δ.

The definitions of $\mathcal{F}_k$ and $\mathcal{F}_k^{-1}$ may look trivial at first. However, they allow us to generalize the propagation algorithm to more complex cases, e.g., filtering out undesired or unknown annotation units or choosing between expanding/collapsing through nested term definitions.

In the case of the $\mathsf{CMM_iM_o}$ strategy, the only difference is that Θ is not simply $\mathcal{F}_k$ applied to $\mathsf{M_o}$, but the result of first executing an ontology alignment operation [11] over $\mathsf{M_i}$ and $\mathsf{M_o}$, and the result is used in stead of $\mathsf{M_o}$. The other two strategies are discussed next.

B. Open Propagation Rules

Open propagation rules propagate annotation units if they do not match another unit under the chosen ontological relations. Section IV-A presented the closed behaviour of the rules ($\mathsf{CMM_o}$ and $\mathsf{CMM_iM_o}$). The version that defines the open behaviour of the rules ($\mathsf{OMM_o}$ and $\mathsf{OMM_iM_o}$) requires a minor modification of the the algorithm in Figure 5. The union step (line 8) of the algorithm is a bit more complicated when adding open rules. The annotations propagated from closed rules should precede the ones from open rules. So, if a given unit was already propagated by a closed rule it should not be considered by any other open rule. However, if the resulting $\mathcal{P}_k$ is empty for a given a in Ω, i.e., if, after comparing a to all b in Θ, $\mathcal{P}_k$ is still empy, then a should be included in the resulting set Δ. The same rationale applies to all b in Θ, requiring a track record if a given b was matched or not. In the latter case, b also should be included in Δ.

The difference between $\mathsf{OMM_o}$ and $\mathsf{OMM_iM_o}$ is akin to the difference between $\mathsf{CMM_o}$ and $\mathsf{CMM_iM_o}$, i.e., in the case of considering M_i, an ontology alignment operation must take place before the application of the algorithm.

Again, it is worth of notice that the open propagation strategies may generate richer annotations, but also erroneous and/or inconsistent ones. This issue is not addressed in this paper and is object for future work.

C. Ontological Relations

The ontological relations used in this paper manipulate the basic elements from an OWL ontology, i.e., the classes, instances and properties. These elements are to be compared to determine which among them will be used as the derived annotation.

The choice of which ontological relations to use in a propagation should be guided by which aspects are useful in a given transformation process. For instance, if class hierarchy relationships are useful, generalization and specialization ontological relations should be used – e.g., considering descendant full compatibility or restricting to only direct subclass compatibility.

We categorize the ontological relations according to which elements from the annotation units are the focus of the comparison. Three categories are defined: classes, instances and properties. Because of space limitation, only five ontological relations are listed – see [12] for additional relations. The first three concern classes category, the fourth instances, and the last one, properties. It is possible to specify many other ontological relations to be used with our mechanism, specially when considering properties. For all ontological relations presented, the annotations units being compared (a and b) are RDF triples *<Subject,Predicate,Object>* and have the form $a =$ `<sa,pa,oa>` and $b =$ `<sb,pb,ob>`.

Class generalization. Relation based on the rdfs:subClassOf construct (defined as part of RDF Schema), which allows replacing general annotation units with more specific ones.

$$\mathcal{R}(a,b) = \begin{cases} \texttt{<sa,pb,ob>} & \\ \qquad \text{if } \texttt{sa } \textit{subClassOf } \texttt{sb} & \\ \texttt{<sb,pb,sa>} & \\ \qquad \text{if } \texttt{sa } \textit{subClassOf } \texttt{ob} & \end{cases}$$

This relation should be read as: given two annotation units $a =$ `<sa,pa,oa>` and $b =$ `<sb,pb,ob>`, generated by a deconstruction function for comparison under *subClassOf*, then the propagated annotation unit is the set composed by `<sa,pb,ob>`, if `sa` is a *subClassOf* `sb`. All other relations are to be read the same way.

Class specialization. This relation is also based on the rdfs:subClassOf construct, since generalization and specialization relations are both described by this construct.

$$\mathcal{R}(a,b) = \begin{cases} \texttt{<sb,pa,oa>} & \\ \qquad \text{if } \texttt{sb } \textit{subClassOf } \texttt{sa} & \\ \texttt{<ob,pa,oa>} & \\ \qquad \text{if } \texttt{ob } \textit{subClassOf } \texttt{sa} & \end{cases}$$

Class complement. Based on the owl:complementOf construct, which represents a class with all properties that are not part of the original class. When two or more units are returned, the result is their conjunction.

$$\mathcal{R}(a,b) = \begin{cases} \texttt{<sa,pa,oa>,<sb,pb,ob>} & \\ \qquad \text{if } \texttt{sa } \textit{complementOf } \texttt{sb} & \\ \texttt{<sa,pa,oa>,<sb,pb,ob>} & \\ \qquad \text{if } \texttt{oa } \textit{complementOf } \texttt{ob} & \end{cases}$$

Instance equivalence. Relation based on owl:sameAs, which states that two instances (represented by different URIs) are, in fact, the same.

$$\mathcal{R}(a,b) = \begin{cases} \texttt{<sa,pa,oa>} & \text{if } \texttt{sa } \textit{sameAs } \texttt{sb} \\ \texttt{<sa,pa,oa>,<sb,pb,ob>} & \text{if } \texttt{sa } \textit{sameAs } \texttt{ob} \\ \texttt{<sa,pa,oa>,<sb,pb,ob>} & \text{if } \texttt{oa } \textit{sameAs } \texttt{sb} \\ \texttt{<sa,pa,oa>} & \text{if } \texttt{oa } \textit{sameAs } \texttt{ob} \end{cases}$$

Property generalization. Relation based on rdfs:subPropertyOf, which favours the use of more specialized properties.

$$\mathcal{R}(a,b) = \begin{cases} \texttt{<sb,pb,ob>} \\ \text{if } \texttt{pb} \textit{ subPropertyOf } \texttt{pa} \text{ and} \\ \texttt{sa} \textit{ equivalentClass } \texttt{sb} \text{ and } \texttt{oa} \textit{ equivalentClass } \texttt{ob} \\ \\ \texttt{<sb,pb,ob>} \\ \text{if } \texttt{pb} \textit{ subPropertyOf } \texttt{pa} \text{ and} \\ \texttt{sa} \textit{ sameAs } \texttt{sb} \text{ and } \texttt{oa} \textit{ sameAs } \texttt{ob} \\ \\ \texttt{<sb,pb,ob>} \\ \text{if } \texttt{pb} \textit{ subPropertyOf } \texttt{pa} \text{ and} \\ \texttt{sb} \textit{ subClassOf } \texttt{sa} \text{ and } \texttt{ob} \textit{ subClassOf } \texttt{oa} \end{cases}$$

Our approach is based on <s,p,o> triple comparison and manipulation. Alternatives to annotation propagation include the use of RDF-S/OWL inference rules or a reasoner to generate the new annotations. Our approach is more direct, enabling us to discuss all the aspects involved in a solution to this problem, e.g., different types of annotation units, annotation structure for data and transformation operations, flexible selection of the propagation mechanism behaviour. Discussing these aspects while considering inference rules or a reasoner would take much more space and is left for future work.

V. Complex Transformation Operations

This section presents how our solution deals with two important aspects of annotation propagation, namely, multiple input/output operations and chained transformation operations.

A. Multiple Input/Output Operations

If a transformation operation has more than one input and/or output, propagation becomes more complicated. We treat each case separately, for clarity sake. For multiple inputs, the idea consists in merging the annotations from each input data set into a single annotation (using ontology alignment), and doing the same for the annotations on the input interface. For multiple outputs, each output is treated as if it were a single output, generating new annotations for each output annotation available.

Multiple Input – Single Output. First, consider a transformation operation with multiple inputs and one output. An example is the overlayMapImages of our running example, which receives two GeoTIFF images as input and produces one GeoTIFF image. Figure 6 illustrates this case. Each input data set (and also, each input expected at the interface) has its associated annotation. To proceed with the propagation, it is necessary to combine the annotations on the input data (M_1, M_2, M_3, in the figure) into a single annotation (M). Likewise, the annotation on the inputs of the interface (M_{i1}, M_{i2}, M_{i3}) must be combined into another single annotation (M_i).

This is accomplished by executing pairwise ontology alignment on the annotations, finally generating the two results (M and M_i) over which the algorithm of Section IV will be applied.

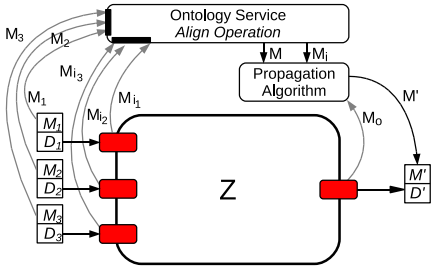

Figure 6. Dealing with multiple inputs.

Single Input – Multiple Output. Here, there are basically two situations. First, the same output is replicated, to be fed into several different operations, which is actually not a multiple output case. Second, several different outputs are generated, representing different results from a transformation operation. In the first case, the propagation mechanism is executed as if for a single output and the result is replicated to each destination of the generated output.

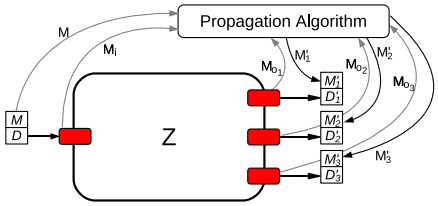

Figure 7. Multiple outputs and multiple propagations.

In the second case, the propagation mechanism must be applied separately for each output of the operation's interface, generating a different annotation to each result, as shown in Figure 7, generating, respectively, $M_1^{'}$, $M_2^{'}$, and $M_3^{'}$. This is equivalent to breaking the operation into several operations, each having the same input interface and input data, but only one of the outputs from the original operation.

Multiple Input – Multiple Output. This combines the strategies of the other cases. The inputs are combined to generate single annotations M and M_i. These combined annotations are then used as if in a single input multiple output case.

B. Composition of Operations

Section IV considered the problem of annotation propagation for one transformation with single input/output, extended in Section V-A to cover transformations with multiple inputs/outputs. The last issue is how to deal with a composite transformation process.

We model processes as workflows and transformations as workflow activities (e.g., the workflow of the running example). The execution of a process is carried out by a *Workflow Management System* (WFMS), which invokes

operations to execute the specified activities. Each activity invocation is followed by an execution of the algorithm of Section IV.

The order in which the activities are executed is determined by the abstract workflow specification. The actual execution is done via workflow instantiations, which preserve the structure from the abstract version and assign data inputs to the workflow and actual operations to each activity. Workflow execution always unfolds into an acyclic sequence of executed steps. Since annotation propagation is executed side by side with activity execution, the existence of cycles in the workflow does not hamper the propagation. Though cycles do not pose an issue, dealing with parallelism (splitting, synchronization, merging) requires attention.

Flow splitting (forking), synchronization, and merging are all controlled by the Workflow Management System. However, a few situations might require special care with respect to the annotation propagation mechanism.

A fork can be treated as a single input/multiple output case, and thus solved by the approach described in Section V-A. For a join, there are three possibilities: (i) it feeds an activity with a single input; (ii) it feeds an activity with multiple inputs (one of which is the result of the join); and, (iii) it feeds an activity with multiple undefined inputs. In the first case, the result of the join is forwarded directly. In case (ii), the join is treated as if it were a virtual activity with multiple inputs and a single output – see Figure 8. Hence, a multiple input strategy can be applied before proceeding with the execution. If the join ends in one activity with multiple inputs and the arriving data is already mapped to the multiple inputs in the workflow, the multiple input strategy can be applied directly. Finally, case (iii), if there is more than one data set arriving for one of the inputs, the strategy for the activity with a single input is used.

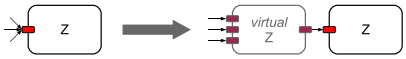

Figure 8. Creation of a virtual join activity.

In a workflow, specification of different sets of ontological relations can be associated with each activity. This allows to customise behaviour for each activity, according to the desired context, when applying the annotation propagation rules.

VI. Revisiting The Example

This section shows the application of our solution, for the example in Figure 1, where the process has two inputs and produces one output. Each operation within the process receives a data set, transforms it and produces an output.

We assume that any instance of the transformation process is steered by a WFMS, which controls the data flow by invoking the activities in the appropriate order and by handling and forwarding the data. Figure 9 illustrates the execution of the workflow in Figure 1, represented at the top. Activity classifyMapImage is highlighted, denoting it is being executed at this specific moment. The bottom of the figure shows that, at execution time, transformation classifyMapImage is instantiated into executing operation A.

Operation A received the data set D and transformed it, generating D'. Annotation propagation is achieved as follows. The WFMS invokes the propagation algorithm, passing as parameters the data annotations M, the input interface annotations M_i, and the output interface annotations M_o. The propagation algorithm executes a set of rules that examines an annotation and produces another annotation. In the figure, it receives annotations M, M_i and M_o and produces M', the derived annotation. The pair (D', M') can then be passed on to the next workflow steps. Ontology access and management take advantage of our Aondê Ontology Web Service [11]. The latter provides access to the most common functions available in ontology toolkits – e.g., find, rank and compare ontologies of interest, create views, and build new ontologies. Ontologies are stored in Ontology Repositories.

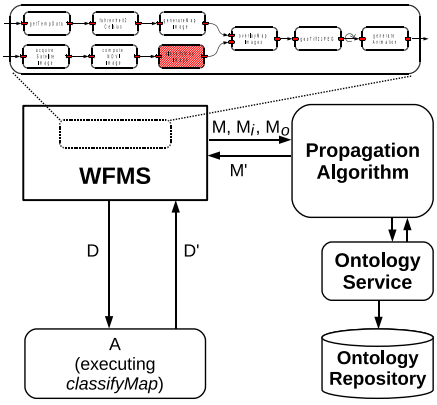

Figure 9. A sample execution.

Derived annotations are thus automatically available at the end of the last step of a data transformation process.

As to the application of the propagation rules, consider again the example of Figure 2, describing the map classification operation, and the propagation strategies in Table 3. For the CMM_iM_o propagation, we have the following (again, omitting the namespaces to improve readability):

$\Omega = \{$ *(id_D, hasEncoding, GeoTIFF-bitmap),*
(id_D, RDFType, NDVI),
(id_D, hasResolution, 250),
(id_D, hasTimestamp, 20010323)
$\}$

$\Theta = \{$ *(id_O, hasEncoding, GeoTIFF-bitmap),*
(id_O, RDFType, ClassifiedVI)
$\}$

$\Delta = \{$ *(id_D', hasEncoding, GeoTIFF-bitmap),*
(id_D', RDFType, ClassifiedVI)
$\}$

The encoding was propagated through the *class equivalence* ontological relation, and the image type was propagated through the *class specialization* ontological relation. The other propagation strategies are similarly applied.

VII. Related Work

This paper is motivated by the increasing need in scientific applications to effectively manage multimedia data, which led to the proposal of a mechanism for annotation propagation. Related work involves therefore examples of multimedia data annotation and propagation proposals.

There is a vast literature on proposals for annotations of multimedia content, frequently motivated by the explosion of multimedia files on the Web. These approaches adopt the expression "annotation propagation" in several ways. A common concern, for instance, is to provide algorithms that annotate videos (or images) in a database, assuming that there is a subset of these videos/images that have already been annotated. Annotation mechanisms, in the case of videos, can concern either an entire video, or specific frames. In the latter case, several kinds of techniques may be used to identify frames of interest within a video, to subsequently apply image annotation solutions. The approach used by [13] to annotate video databases, to cite but one paper, is typical of this kind of concern. In such annotation studies, the notion of "annotation propagation" refers to using a subset of annotated images (or videos) as a basis to learn how to annotate the entire set. Our work is not concerned with this kind of propagation – rather, we assume that all relevant multimedia data sets used as the input of a process have already been annotated (e.g., using approaches like the one proposed by [13]).

Metadata have been used in several annotation contexts [5, 14, 15] including multimedia [16–19]. Particularly, they have proven useful in establishing means of assigning descriptions to multimedia content, its execution and user interaction environments. The works of [20, 21] argue that descriptions of (i) multimedia content and (ii) users' preferences related to multimedia content are essential to achieve what they call *universal multimedia access* (UMA). In UMA, the multimedia content should be available anywhere and anytime, possibly using content adaptation to achieve this. A similar vision is shared by [22] which discusses the issues arising from ambient intelligence; by [23] which describes how user input, sensor readings, available media and software are needed to achieve such an environment; and by [24] which analyses this scenario for distance learning. This view can be generalised to context description, using metadata to describe the whole environment along with the content manipulated.

However, there have been some difficulties in using metadata. The work of [25] analyses the trade-offs of using metadata in several scenarios, including appropriate uses where the environment is data-driven and/or requires analytical decision making; and less appropriate uses when considering more intuitive or politically charged environments. It also points out the importance of metadata quality and completeness in the successful use of metadata. Another problem with metadata-based strategies is the lack of user motivation to go through the tedious process of creating metadata. For instance, Bulterman [26] discusses these issues concerning MPEG-7 and its commercial motivation as a reason for its relative success. These works stress the need for more automation on the generation of metadata, which is the main concern of our solution.

Other papers concern the adaptation of metadata to be used on the Web, often following standards of the Semantic Web [27–32]. The application of Semantic Web standards to describe multimedia content involves the production of semantic annotations and thus could profit from our annotation propagation solution. On another related front, work with Semantic Web Services (e.g., [33, 34]) provide the possibility of semantic annotations on interfaces of operations, thereby meeting a requirement of our solution (i.e., describing the interfaces of operations). The Multimedia Annotation Interoperability Framework [35] also considers the problem of describing transformation operations focusing on multimedia content.

One use of the term "annotation propagation" [36–40] regards the automated update of annotations in a hierarchy of already annotated objects (e.g., if a group receives an annotation, all entities from that group receive it as well; or if an annotation is corrected, all related annotations get corrected as well). An example of this kind of work is found in [41], which studies how annotations on database tuples are propagated to views on these tuples - and, subsequently, how view updates on annotated tuples reflect on database annotations. Their study is restricted to a specific kind of views (key-preserving SPJ views), for which database literature has shown they have predictable behaviour during database updates. In other words, this kind of annotation propagation is studied for very specific transformations and conditions.

Our use of the term, however, regards data transformations and their impact on the annotations. To the best of our knowledge, there are two approaches in which the notion of "annotation propagation" is the same as ours. The first solution is presented in [7]. Their solution is placed in a data warehouse environment, with the annotations stored in extra (data) columns. The paper presents rules for propagating these annotation fields to answer queries. Rule implementation is based on pSQL, an extension of the SQL query language that supports a *propagate* clause to enable the application of propagation schemes. The solution of [7] is restricted to be used within databases and data warehousing environments, being limited by the query language. It only considers fine-grained annotations, in an item-by-item basis. Our solution, on the other hand, can be applied to any kind of transformation and allows any granularity on the annotations, provided that both the transformations and the data are described with semantic annotations.

The second proposal [8] attacks the annotation propa-

gation problem by taking advantage of database schema mapping compositions. Their solution is restricted to relational algebra operations and is applicable to sequences of transformations modelled as workflows. Similar to [8], we also annotate content using ontologies, but without restriction to database system environments.

In addition, the work of [10] proposes a mechanism for tracing transformations for data warehouses. They present aspects of dealing with general transformations, including dealing with multiple inputs/outputs and general sequences (or graphs) of data transformations. These issues were also dealt with by us in our mechanism, but in the context of annotation propagation.

As far as we know, ours is the only solution that considers general data transformation operations. It is also the only one to take into account both previous data descriptions and operation interface descriptions.

VIII. Concluding Remarks

This paper presented a new approach to propagate annotations on multimedia data sets. A solution to the annotation propagation problem offers several advantages, such as: (i) lessening annotation efforts, (ii) decreasing the loss of information along the transformation process, (iii) documenting data origins (for traceability and provenance), and (iv) providing quality information. This paper focused on the first two issues.

Our approach allows combining content-based metadata with contextual information provided by ontologies. Solutions to the annotation propagation problem have so far been restricted to database operations. We, instead, encompass general transformation operations. Rather than having to consider the operations themselves, our propagation mechanism requires only the annotations on the input/output interfaces of the operation. Moreover, the classification of propagation strategies introduced in the paper were reflected in open and closed propagation rules.

Our solution is general and can be applied in service-based environments. It can also be applied in more specific or controlled environments, such as a database system or a digital library system. Our mechanism can also be extended by increasing the number of ontological relations. This allows tailoring annotation propagation behaviours, permitting new specific relationships to be considered.

As future work, we intend to investigate annotation propagation considering particular outcomes of each data manipulation function, e.g., content-based annotation propagation that generates annotations similar to those of [42, 43]. For instance, parts of the output could be annotated individually with different annotations. This means that the propagation mechanism could generate different annotations depending not only on the previous annotations and operation interfaces, but also on the resulting data set.

Another aspect that is worth investigating is the relationship between annotation propagation and provenance management techniques. An annotation propagation mechanism such as ours can be used as part of provenance related efforts. For instance, it can propagate original provenance information. Moreover, it can help to keep track of data transformation processes, thereby semantically enhancing process provenance descriptions. Efforts towards connecting domain and provenance ontologies – e.g., [44] – help enabling such combination.

We also plan to investigate how to deal with the possibility of errors in open propagation rules. The use of more specific ontology inference mechanisms could help evaluate the possibility of errors. Furthermore, execution logs with human validation could be used to automatically create extra annotations. This helps in decreasing the number of errors. Another possibility would be to use RDF(S)/OWL inference rules or a reasoner instead of the propagation rules. This might require additional parametrization – e.g., letting users assign priorities to rules.

Acknowledgements

This work was supported by FAPESP (grant 2004/14052-3) and partially financed by the FAPESP–Microsoft Research Virtual Institute (eFarms project) and CNPq.

References

[1] N. Dawes, K. Kumar, S. Michel, K. Aberer, and M. Lehning, "Sensor Metadata Management and its Application in Collaborative Environmental Research," in *Proc. 4th IEEE Int. Conf. on eScience*, 2008, pp. 143–150.

[2] K. Böhm and T. C. Rakow, "Metadata for Multimedia Documents," *SIGMOD Record*, vol. 23, no. 4, pp. 21–26, 1994.

[3] E. Duval, W. Hodgins, S. Sutton, and S. L. Weibel, "Metadata Principles and Practicalities," *D-Lib Magazine*, vol. 8, no. 4, 2002.

[4] M. L. Kherfi and D. Ziou, "Image Collection Organization and Its Application to Indexing, Browsing, Summarization, and Semantic Retrieval," *IEEE Transactions on Multimedia*, vol. 9, no. 4, pp. 893–900, 2007.

[5] M. Agosti and N. Ferro, "A Formal Model of Annotations of Digital Content," *ACM Transactions on Information Systems*, vol. 26, no. 1, p. 3, 2007.

[6] K. Haase, "Context for Semantic Metadata," in *Proc 12th ACM International Conference on Multimedia*, 2004, pp. 204–211.

[7] D. Bhagwat, L. Chiticariu, W. Tan, and G. Vijayvargiya, "An annotation management system for relational databases," *The VLDB Journal*, vol. 14, no. 4, pp. 373–396, 2005.

[8] S. Bowers and B. Ludäscher, "A Calculus for Propagating Semantic Annotations Through Scientific Workflow Queries," in *EDBT Workshops*, 2006, pp. 712–723.

[9] G. Z. Pastorello Jr, J. Daltio, and C. B. Medeiros, "Multimedia Semantic Annotation Propagation," in *Proc. of the 1st IEEE Int. Workshop on Data Semantics for Multimedia Systems and Applications (DSMSA) – 10th IEEE Int. Symposium on Multimedia*, 2008, pp. 509–514.

[10] Y. Cui and J. Widom, "Lineage tracing for general data warehouse transformations," *The VLDB Journal*, vol. 12, no. 1, pp. 41–58, 2003.

[11] J. Daltio and C. B. Medeiros, "Aondê: An Ontology Web Service for Interoperability across Biodiversity Applications," *Information Systems*, vol. 33, no. 7-8, pp. 724–753, 2008.

[12] G. Z. Pastorello Jr, J. Daltio, and C. B. Medeiros, "An Annotation Propagation Mechanism for Multimedia Content," Institute of Computing, University of Campinas, Technical Report IC-08-017, August 2008.
[13] J. Tang, X.-S. Hua, G.-J. Qi, Y. Song, and X. Wu, "Video Annotation Based on Kernel Linear Neighborhood Propagation," *IEEE Transactions on Multimedia*, vol. 10, no. 4, pp. 620–628, 2008.
[14] W. Cathro, "Metadata: An Overview," in *Standards Australia Seminar: Matching Discovery and Recovery*, 1997, http://www.nla.gov.au/nla/staffpaper/cathro3.html (as of Apr 2008).
[15] A. Sen, "Metadata management: past, present and future," *Decision Support Systems*, vol. 37, no. 1, pp. 151–173, 2004.
[16] Moving Picture Experts Group (MPEG) – ISO/IEC-JTC1, "ISO/IEC15938 (parts 1–10) Multimedia content description interface," http://www.iso.org/iso/search.htm?qt=15938&sort=rel&type=simple&published=true (as of Sep 2008).
[17] H. Kosch, L. Boszormenyi, M. Doller, M. Libsie, P. Schojer, and A. Kofler, "The Life Cycle of Multimedia Metadata," *IEEE Multimedia*, vol. 12, no. 1, pp. 80–86, 2005.
[18] K. Falkovych and F. Nack, "Context aware guidance for multimedia authoring: harmonizing domain and discourse knowledge," *Multimedia Systems*, vol. 11, no. 3, pp. 226–235, 2006.
[19] R. Goularte, M. G. C. Pimentel, and E. S. Moreira, "Context-aware support in structured documents for interactive-TV," *Multimedia Systems*, vol. 11, no. 4, pp. 367–382, 2006.
[20] P. Beek, J. R. Smith, T. Ebrahimi, T. Suzuki, and J. Askelof, "Metadata-driven Multimedia Access," *IEEE Signal Processing Magazine*, vol. 20, no. 2, pp. 40–52, 2003.
[21] F. Pereira and I. Burnett, "Universal Multimedia Experiences for Tomorrow," *IEEE Signal Processing Magazine*, vol. 20, no. 2, pp. 63–73, 2003.
[22] E. Aarts, "Ambient Intelligence: A Multimedia Perspective," *IEEE MultiMedia*, vol. 11, no. 1, pp. 12–19, 2004.
[23] S. P. Stenton, R. Hull, P. M. Goddi, J. E. Reid, B. J. C. Clayton, T. J. Melamed, and S. Wee, "Mediascapes: Context-Aware Multimedia Experiences," *IEEE Multimedia*, vol. 14, no. 3, pp. 98–105, 2007.
[24] M. Almaoui, A. Kushki, and K. N. Plataniotis, "Metadata-driven multimedia transcoding for distance learning," *Multimedia Systems*, vol. 12, no. 6, pp. 505–520, 2007.
[25] G. Shankaranarayanan and A. Even, "The Metadata Enigma," *Communications of the ACM*, vol. 49, no. 2, pp. 88–94, 2006.
[26] D. C. A. Bulterman, "Is It Time for a Moratorium on Metadata?" *IEEE Multimedia*, vol. 11, no. 4, pp. 10–17, 2004.
[27] G. Stamou, J. Ossenbruggen, J. Z. Pan, G. Schreiber, and J. R. Smith, "Multimedia Annotations on the Semantic Web," *IEEE Multimedia*, vol. 13, no. 1, pp. 86–90, 2006.
[28] H. Wang, S. Liu, and L.-T. Chia, "Image retrieval with a multi-modality ontology," *Multimedia Systems*, vol. 13, no. 5-6, pp. 379–390, 2008.
[29] R. Arndt, R. Troncy, S. Staab, L. Hardman, and M. Vacura, "COMM: Designing a Well-Founded Multimedia Ontology for the Web," in *Proc. 6th Int. Semantic Web Conf. and 2nd Asian Semantic Web Conf.*, 2007, pp. 30–43.
[30] M. Ferecatu, N. Boujemaa, and M. Crucianu, "Semantic interactive image retrieval combining visual and conceptual content description," *Multimedia Systems*, vol. 13, no. 5-6, pp. 309–322, 2008.
[31] R. Garcia and O. Celma, "Semantic Integration and Retrieval of Multimedia Metadata," in *Proc. 5th International Workshop on Knowledge Markup and Semantic Annotation*, 2005.
[32] M. Naphade, J. R. Smith, J. Tesic, S. Chang, W. Hsu, L. Kennedy, A. Hauptmann, and J. Curtis, "Large-scale Concept Ontology for Multimedia," *IEEE Multimedia*, vol. 13, no. 3, pp. 86–91, 2006.
[33] OWL-S (The DAML Program), "OWL-based Web Service Ontology," http://www.daml.org/services/owl-s/ (as of Apr 2008).
[34] W3C, "Semantic Annotations for Web Service Description Language (SAWSDL)," http://www.w3.org/2002/ws/sawsdl (as of Apr 2008).
[35] W3C Multimedia Semantics Incubator Group, "Multimedia Annotation Interoperability Framework," http://www.w3.org/2005/Incubator/mmsem/XGR-interoperability/ (as of September 2008).
[36] P.-H. Luong and R. Dieng-Kuntz, "A Rule-Based Approach for Semantic Annotation Evolution," *Computational Intelligence*, vol. 23, no. 3, pp. 320–338, 2007.
[37] G. Langs, P. Peloschek, R. Donner, and H. Bischof, "Annotation Propagation by MDL Based Correspondences," in *Proc. Computer Vision Winter Workshop 2006*, 2006.
[38] F. Kang and M. R. Naphade, "A Generalized Multiple Instance Learning Algorithm for Iterative Distillation and Cross-Granular Propagation of Video Annotations," in *Proc. IEEE International Conference on Image Processing*, 2007, pp. II–205–208.
[39] W. Ku, M. S. Kankanhalli, and J.-H. Lim, "Metadata management, reuse, inference and propagation in a collection-oriented metadata framework for digital images," in *Advances in Multimedia Modeling: 13th Int. Multimedia Modeling Conf*, 2007, pp. 145–154.
[40] E. Onasoglou and P. Daras, "Semantic force relevance feedback, content-free 3d object retrieval and annotation propagation: bridging the gap and beyond," *Multimedia Tools and Applications*, vol. 39, no. 2, pp. 217–241, 2008.
[41] G. Cong, W. Fan, and F. Geerts, "Annotation Propagation Revisited for Key Preserving Views," in *Proc. of the 15th ACM Int. Conf. on Information and Knowledge Management (CIKM)*, 2006, pp. 632–641.
[42] N. Vasconcelos, "From Pixels to Semantic Spaces: Advances in Content-Based Image Retrieval," *IEEE Computer*, vol. 40, no. 7, pp. 20–26, 2007.
[43] J. Yuan, J. Li, and B. Zhang, "Exploiting spatial context constraints for automatic image region annotation," in *Proc. 15th International Conference on Multimedia*, 2007, pp. 595–604.
[44] S. S. Sahoo, A. Sheth, and C. Henson, "Semantic provenance for escience: Managing the deluge of scientific data," *IEEE Internet Computing*, vol. 12, no. 4, pp. 46–54, 2008.

G. Z. Pastorello Jr, Post Doctoral Fellow at University of Alberta, receiveid his PhD (2008), MSc (2005) and BSc (2003) in Computer Science from University of Campinas.

J. Daltio, Systems Analyst at Trópico, received her MSc (2007) in Computer Science from University of Campinas and BSc (2005) from University of Viçosa.

C. B. Medeiros is full professor of Computer Science at the Institute of Computing, University of Campinas, Brazil. Her research is centered on scientific databases and eScience.