INPE ePrint: sid.inpe.br/yolanda/2004/07.08.17.41 v1 2004-07-16

Procedência de dados: teoria e aplicações ao processamento de imagens

JULIANA CRISTINA BRAGA, GERALD JEAN FRANCIS BANON
INPE - Instituto Nacional de Pesquisas espaciais
São José dos Campos - SP - Brasil
{juliana, banon}@dpi.inpe.br

Abstract. The provenance is a complementary data documentation. It contains a description of "how", "when", "where", "why" the data were obtained and "who" obtained it. The provenance includes not only the origin of the data but also the description of the processes that produces it. Data provenance might be very helpful in image processing to reproduce images, save disk space, increasing speed transmission, protect the copyright and retrieve images. In our work, we have developed a Web processing system that capture and manage image provenance. This systems is integrated with the UR*Lib* digital library. Some preliminary tests have shown the feasibility and efficiency of the proposed data provenance management system.

1 Introdução

A procedência de dados é a documentação complementar a um dado que contém a descrição de "como", "quando", "onde", "porque" ele foi obtido e "quem" o obteve. Ela inclui não só a origem do dado (identificação, responsável pelo dado, data de criação), mas também os processos aplicados a ele (algoritmos e seus respectivos parâmetros) [11].

Em uma abordagem diferente proposta por [6] o termo procedência é utilizado para fazer referência a informação de onde o dado vem e o termo "pedigree" para fazer referência ao histórico de como o dado foi produzido. Consideramos essa abordagem válida, porém muito específica e por isso admitimos em nosso trabalho que origem e pedigree fazem parte fazem parte da procedência.

Observamos na literatura que a procedência de dados é um campo novo e por isso ainda existe muita divergência sobre sua nomenclatura. Além disso, o conceito de procedência varia de acordo com as comunidades científicas e o tipo de aplicação. Algumas área já descobriram o potencial da procedência de dados. Com exemplo podemos citar Sistemas de Informação, Banco de Dados, Astronomia, Bioinformática, Ciências Químicas.

Neste trabalho, tratamos da procedência de dados oriundos de processos computacionais, mais especificamente processamento de imagens. Sendo assim, a seguir identificamos os benefícios da procedência de dados em processamento de imagens [1].

• Reproduzir imagens: muitas vezes, temos a necessidade de repetir um processamento de imagens existente. Um exemplo, é quando lemos um artigo científico e queremos reproduzir o processamento nele descrito. Através da procedência das imagens, podemos obter detalhes do processamento que elas sofreram e repetir o processo com eficiência.

• Adaptar processos existentes: podemos repetir um processamento bem sucedido com novos dados de entrada, ou então testar a influência de um parâmetro sobre o resultado final. Isto e possível consultando a procedência das imagens, ajustando alguns parâmetros e executando novamente o processo. Esta é uma maneira de evitar a repetição de esforços de pesquisa e facilitar sua continuidade.

• Economizar espaço em disco: certas imagens podem ser vistas como resultados intermediários dentro de uma cadeia de processamento. Neste caso, se existir a procedência dessas imagens intermediárias, elas podem ser apagadas depois de finalizado o último tratamento que as envolve, deixando assim mais espaço para o armazenamento de futuros tratamentos. Caso necessitarmos usar uma dessas imagens novamente, basta consultarmos sua procedência para saber como a mesma foi obtida e gerá-la outra vez.

• Aumentar a velocidade de transmissão: com as tecnologias atuais, a troca de dados através de Internet cresce a cada dia. Em alguns casos, a transmissão da procedência ao invés da imagem pode ser sensivelmente mais rápida. Ou seja, a reprodução das imagens através de sua procedência pode ser mais eficaz do que sua transferência.

• Proteger direitos de propriedade: a procedência de imagens possui a informação de quem a gerou, sendo assim ela permite auxiliar no tratamento de questões relacionadas ao direito de propriedade intelectual de um dado.

• Auxiliar na busca e recuperação de imagens na Web: a procedência pode guardar de forma organizada informações a respeito das características da imagem. Mecanismos de busca da Web poderão usufruir disso para encontrar imagens através de seu conteúdo.

São grandes os benefícios da procedência para processamento de imagens porém para usufruí-los é

fundamental que a procedência seja bem gerenciada. Ser bem gerenciada significa ser criada de forma adequada, ser armazenada em meios seguros, ser bem disseminada e ser facilmente encontrada e aproveitada.

Acreditamos que somente um sistema computacional, com a mínima intervenção do usuário é capaz de gerenciar adequadamente a procedência (desde criação até publicação) . Porém, ainda existem alguns desafios computacionais para o gerenciamento da procedência. Esses, estão relacionados com a criação, armazenamento, publicação, busca, recuperação e preservação da procedência e das imagens.

Na maioria dos processamentos de imagens, a procedência não existe, e quando existe é criada manualmente ou automaticamente por softwares de processamento (arquivos de registro ou "log"). Se feita manualmente, é incompleta, apresenta erros e sem padronização. Quando gravada automaticamente é de difícil compreensão e em formato proprietário. Esses fatores dificultam o acesso e a manipulação da procedência por sistemas computacionais.

Se a procedência for armazenada adequadamente, em meios seguros e confiáveis, a chance de sua preservação aumenta, caso contrário sua existência é comprometida. Atualmente o armazenamento de procedências de imagens é gerenciado somente pelo responsável pelo processamento. Este, pode armazena-la adequadamente mas também apagá-la ou até mesmo esquecer de sua existência.

Sabemos que conduzir uma pesquisa requer disseminação e troca de dados assim como a publicação de suas conclusões. Ou seja, a divulgação da procedência é tão importante quanto sua existência. Em muitos casos os pesquisadores registram a procedência de suas imagens mas raramente as disseminam. Assim, a procedência torna-se útil apenas para o próprio pesquisador sem que outros grupos de pesquisa aproveitem de seus benefícios.

Como vimos, uma das grandes vantagens da procedência é podemos gerar uma imagem novamente, ou seja reproduzir uma imagem, utilizando sua procedência. Porém, em processamento de imagem, a obtenção da imagem resultante (imagem final) depende do processamento de outras imagens (imagens originais). Nesses casos, para reproduzir a imagem final, somente sua procedência não é suficiente, seria necessário também as imagens originais. Sendo assim, a preservação das imagens originais é tão importante quanto a preservação da procedência

Em vista dos grandes benefícios da procedência de imagens e dos desafios em seu gerenciamento, os objetivos gerais dessa tese são:

• Fundamentar a procedência de dados para orientar a implementação de seu gerenciamento computacional.

• Implementar um protótipo para o gerenciamento da procedência.

• Testar o protótipo em processamento de imagens..

2 Fundamentos da Procedência

São três os aspectos fundamentais para que a procedência de dados seja entendida e consequentemente bem gerenciada.

O primeiro aspecto, Modelo, mostra as informações que a procedência deve conter, com ênfase na linhagem dos dados. O segundo aspecto, Estrutura, formaliza uma estrutura adequada para procedência. O terceiro aspecto, Armazenamento, propõe guardar a procedência em meios seguros que cuidam de sua integridade, preservação, persistência, assim como dos direitos de propriedade, busca e recuperação.

Nos tópicos seguintes abordaremos os três aspectos fundamentais da procedência de dados.

2.1 Modelo

O Modelo da procedência refere-se ao tipo de informação que deve estar contida na mesma. Consideramos que a procedência de um dado deve conter informações suficientes para que pessoas não envolvidas na geração desse dado possam reproduzi-lo através de sua procedência. Sendo assim, o modelo de procedência deve conter dois componentes principais: a origem e a linhagem do dado. Devido a complexidade, um tópico específico sobre linhagem de dados será abordado.

A origem do dado contém, no mínimo, o identificador, data de criação do dado, e o responsável pela criação do mesmo.

A linhagem refere-se à como o dado foi processado ou seja, a relação de todas as funções e parâmetros aplicados ao processamento que o dado sofreu.

Linhagem de Dados

A linhagem é especialmente importante para a procedência de imagens. Pouco adianta saber a origem de uma imagem sem saber a forma como ela foi processada. A linhagem é utilizada para auxiliar na preservação de imagens, evitar repetição de esforços na tentativa de dar continuidade ou refazer processamentos existentes.

A linhagem deve ser gerada automaticamente por programas de computadores para permitir eficácia e precisão em sua manipulação. Para que sua implementação e gerenciamento computacional sejam coerentes é necessário a sua formalização.

A seguir introduzimos os novos conceitos de árvore de linhagem, linhagem para um dado e linhagem de um nome reservado. Esses conceitos foram baseados na teoria de linguagens formais [9].

Seja B o conjunto de frases de uma certa linguagem de programação. Essas frases podem ser geradas por uma gramática de livre-contexto $G = (V, \Sigma, P, S)$, tem-se $B = L(G)$. Exemplificando-se:

Σ = {1, 2, 3, 4, 5, 6, , ∧, ∨, (,), ~} (alfabeto terminal)

N = {<element>, <max>, <min>, <inv>, <sentence>}
(conjunto de variáveis)

$V = \Sigma + N$ (vocabulário)

S = <sentence> (símbolo inicial)

O conjunto das regras de produção, P, para geração das frases, escritas na Forma de Backus-Naur (BNF), é composto por:

<element> ::= 1 | 2 | 3 | 4 | 5 | 6 (regras 1 a 6)

<max> ::= ∨ (<sentence> <sentence>) (regra 7)

<min> ::= ∧ (<sentence> <sentence>) (regra 8)

<inv> ::= ~ (<sentence>) (regra 9)

<sentence> ::= <element> | <max> | <min> | <inv>
(regras 10 a 13)

As seguintes expressões são exemplos de frases nesta linguagem:

2

~(2)

3

∧(~(2) 3)

A última frase foi gerada aplicando os seguintes passos:

<sentence> → <min> (aplicação da regra 12)

<sentence> → ^(<sentence> <sentence>)
(aplicação da regra 8)

<sentence> → ^ (<inv> <sentence>)
(aplicação da regra 13)

<sentence> → ^ (~(< sentence>) <sentence>)
(aplicação da regra 9)

<sentence> → ^ (~(<element>) <sentence>)
(aplicação da regra 10)

<sentence> → ^ (~(2) <sentence>) (aplicação da regra 2)

<sentence> → ^ (~(2) <element>) (aplicação da regra 10)

<sentence> → ^ (~(2), 3) (aplicação da regra 3)

A árvore de derivação referente as regras de produção para obter a frase ^(~(2), 3) é representada na Figura 1:

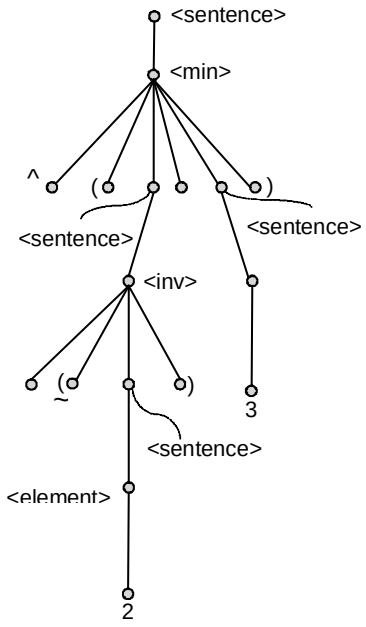

Figura 1 - Árvore de derivação referente as regras de produção para obter a frase ^(~(2) 3)

A partir deste ponto, alguns conceitos novos serão introduzidos.

Percorrendo a árvore de derivação debaixo para cima podemos gerar a árvore de linhagens. Na árvore de linhagens cada nó é rotulado por uma frase que representa a linhagem para um certo dado (ver abaixo definição de linhagem para um).

A árvore de linhagem é obtida identificando na árvore de derivação os nós <sentence> (símbolo inicial). A cada um destes nós associa-se de forma biunívoca um só nó na árvore de linhagem. Cada nó é rotulado pela concatenação das folhas na árvore de derivação percorrendo essa árvore da esquerda para direita.

A Figura 2 mostra a árvore de linhagem obtida a partir da árvore de derivação da Figura 1.

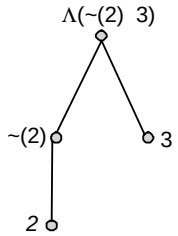

Figura 2 - Árvore de linhagem da frase ^(~(2) 3) gerada a partir da sua árvore de derivação.

Para formalizar o relacionamento entre, linhagem, dados e nomes reservados considera-se dois mapeamentos.

O primeiro mapeamento chamado de operador exec, associa cada frase da linguagem um único dado. Em outros termos, o operador exec é um mapeamento de *B* em *C*, onde *C* é o conjunto de dados.

Se $e \in C$, então por definição, uma frase *s* de *B* é *uma linhagem para um dado e*, ou de forma abusiva *linhagem de e*, se e somente se, exec(*s*) = *e*.

A Figura 3 ilustra o primeiro mapeamento

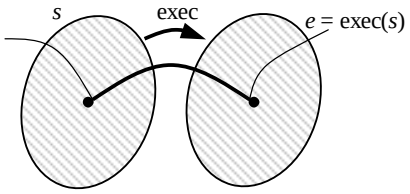

Figura 3 - Mapeamento exec.

Retornando ao exemplo anterior, admite-se que $C = \{1, 2, 3, 4, 5, 6\}$ e que as variáveis: <element>, <max>, <min> e <inv> são processadas da seguinte maneira: para qualquer $e, a, b \in C$,

exec(e) = e

exec($\vee$ (a b)) = max{a, b}

exec($\wedge$ (a b)) = min{a, b}

exec($\sim$(e)) = 6 - e

Desta forma o resultado do processamento do exemplo da Figura 2 fica:

exec(2) = 2

exec(3) = 3

exec($\sim$(2)) = 4

exec($\wedge$($\sim$(2), 3)) = 3

Substituindo os rótulos da árvore de linhagem da Figura 2 pelo resultado da aplicação do comando exec obtemos a árvore de dados da Figura 4.

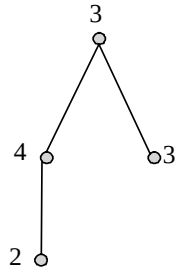

Figura 4 - Árvore de dados

Chamamos de dados originais os dados representados pelas folhas da árvore de dados. Chamamos de dado resultante o dado representado pela raiz da árvore de dados. Chamamos de dados intermediários os dados representados pelos nodos internos incluindo a raiz.

Ao aplicar o comando exec na linhagem do dado resultante, conseguimos recuperar a partir dos dados originais os dados intermediários e o dado resultante. Quando dizemos recuperar um dado significa criá-lo novamente. É importante observar que para recuperação de dados através de sua linhagem é indispensável a preservação dos dados originais.

Considera-se agora o segundo mapeamento. O segundo mapeamento estabelece o critério para se dar nomes reservados às frases de uma dada linguagem. Os nomes serão úteis para referenciar frases e armazená-las.

Criado uma frase, dá-se um nome reservado, isto é, que nunca foi usado antes e que nunca mais será usado. Para isto, escolhe-se de forma apropriada um nome dentro de um conjunto infinito de nomes. Seja linh o mapeamento do conjunto *A* de nomes reservados para o conjunto *B* de frases.

Seja $n \in A$ então por definição, uma frase *s* de *B* é a *linhagem de um nome reservado n* se e somente se s = linh(*n*).

Desta forma, a cada nome escolhido corresponderá uma única linhagem. Procurar-se-á, na medida do possível, que nomes diferentes correspondam a linhagens diferentes mas isto não é imprescindível. Neste caso, o mapeamento linh seria injetora ("onde-to-one").

Um nome para uma linhagem *s* servirá de endereço para o local onde estará sendo armazenado *s* e o resultado de sua execução.

A

Figura 5 mostra a composição dos dois mapeamentos introduzidos anteriormente.

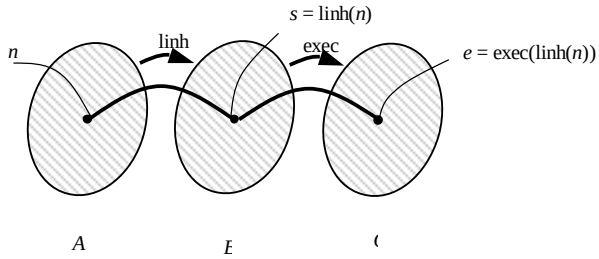

Figura 5 - Composição dos mapeamentos linh e exec.

Definido o mapeamento linh é possível dar nomes às frases consideradas no exemplo desta sessão. O diagrama de blocos da Figura 6 mostra a alocação de nomes para o exemplo mostrado na Figura 2.

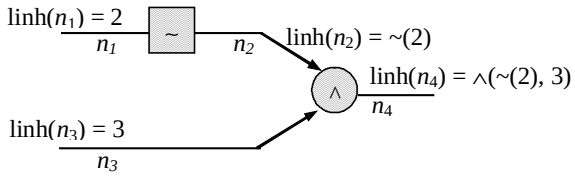

Figura 6 - Alocação de nomes

Observa-se que linh(n_4) pode ainda ser escrito como:

linh(n_4) = ∧(linh(n_2) linh(n_3))

onde

linh(n_3) = 3

linh(n_2) = ~(linh(n_1))

linh(n_1) = 2

Por abuso de notação escreveremos linh(n) como sendo n. Então a linhagem de n_4 ficaria:

n_4 = ∧(n_2, n_3)

onde

n_3 = 3

n_2 = ~(n_1)

n_1 = 2

Isto é, a partir dos nomes n_1, n_2, n_3 seria possível "calcular" (efetuando substituição) a linhagem de n_4.

Estrutura

Para o gerenciamento computacional da procedência de dados, é importante que o modelo de procedência proposto (origem e linhagem) obedeça uma estrutura padrão e organizada.

A estrutura da procedência é a forma ou maneira como a informação contida na procedência deve estar organizada. Por ser um dado que descreve outro dado, podemos representar a procedência como metadados [8]. Para que possamos descrever as informações contidas em metadados de forma estruturada e padronizada devemos seguir um modelo ou esquema de metadados.

Existem diferentes esquemas de metadados para descrever dados. Como o objetivo de nosso trabalho é propor soluções genéricas e apesar de existirem esquemas de metadados para descrição de imagens, vamos utilizar o Dublin Core (DC) como modelo para estruturar a procedência de imagens. Em suma, sugerimos escrever a procedência de dados em XML (eXtensible Markup Language) seguindo o esquema DC.

Armazenamento

Sugerimos guardar a procedência e os dados numa biblioteca digital. São grandes as vantagens em armazenar a procedência e os dados em Bibliotecas Digitais dentre as quais citamos:

- Os dados são armazenados em base de dados distribuída.

- Manutenção da Integridade referencial dos dados: a biblioteca digital garante que a referência física entre a procedência e dados não são perdidos.

- Informação facilmente compartilhada: os dados depositados numa biblioteca digital são disponibilizados na Internet, o que possibilita uma ótima disseminação da procedência.

- Informação sempre disponível: a qualquer momento podemos consultar e recuperar dados e sua procedência, uma vez que os acervos ficam disponíveis 24 horas.

- Acesso persistente: qualquer que seja o acervo local onde foi depositado a procedência, seu acesso é persistente, ou seja ele sempre estará disponível mesmo que tenha mudado de acervo local.

- Respeitar o direito moral e patrimonial do autor e detentor do documento: armazenar um documento numa biblioteca digital é um ato de publicação [2]. Sendo assim, a biblioteca digital, juntamente com a procedência do dado, permitem identificar com confiança quem é o detentor do direito moral (autor) e o detentor do direito patrimonial (pessoa ou instituição que mantém o acervo local) dos processamentos descritos na procedência.

A biblioteca escolhida para armazenar a procedência de dados foi a UR*Lib* - "Uniform Repositories for a Library" (Repositórios Uniformes para uma Biblioteca). [2]. A UR*Lib* vem sendo desenvolvida desde 1995 por Gerald Jean Francis Banon. Ela hospeda a Memória Técnico Científica do INPE [3]. Uma vez depositada a procedência na UR*Lib*, o UR*Lib*Service se encarrega da disseminação, busca e recuperação da procedência de dados.

3 Protótipo

Implementamos um protótipo baseado nos fundamentos da procedência. O Protótipo é um sistema de processamento que gerencia a procedência dos dados por ele processado. O gerenciamento envolve: criação, armazenamento, busca e recuperação da procedência, preservação dos dados originais e reprodução dos dados intermediários.

Chamamos o protótipo de Sistema Gerenciador Procedência de Dados (SGPD). O SGPD é um sistema Cliente/Servidor desenvolvido em Python 2.1. O protocolo de comunicação usado para troca de dados entre navegador (cliente) e SGPD (servidor) é o Hypertext Transfer Protocol (HTTP). O Servidor HTTP utilizado é o Apache. O SGPD foi instalado em uns dos acervos da UR*Lib* e pode ser acessado no endereço: `http://hermes.dpi.inpe.br:1910/col/dpi.inpe.br/juliana/2003/08.18.12.28/doc`

Ao acessar o SGPD o usuário previamente cadastrado entra em sua Área de trabalho. A Área de trabalho pode conter várias Sessões, cada uma delas relativas a um processamento diferente. A Figura 7 mostra uma Sessão pertencente a uma determinada Área de Trabalho.

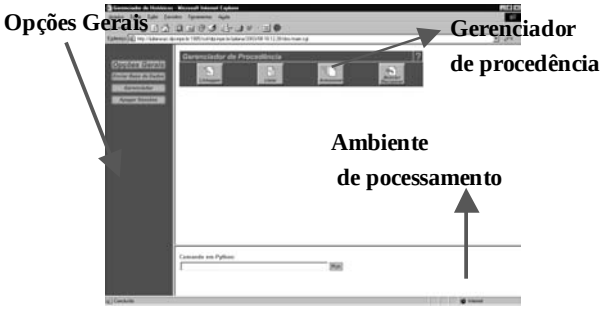

Figura 7 - Área de trabalho do SGPD.

Observamos na Figura 7 que no canto superior localizam-se os comandos relativos ao gerenciamento da procedência, no canto inferior é a área reservada para entrada dos comandos de processamento. Ao lado esquerdo estão as opções de sistema.

4 Testes

Para testar o uso do SGPD em processamentos de imagens refizemos o processamento realizado no artigo "Restauração de imagens NOAA por Morfologia Matemática" apresentado no VI Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens [4]. O artigo utiliza Morfologia Matemática para reduzir o efeito de listas em imagens NOAA.

Para reproduzir o processamento relatado no artigo em estudo, utilizamos o SGPD e a caixa de ferramenta em morfologia matemática desenvolvida em Python 2.2 [5,10].

A Figura 8 mostra a imagem original com o efeito de listras. A imagem original contém uma linha horizontal causada por um defeito no sensor no momento da captura da imagem.

A partir da imagem original iniciamos o processamento descrito no referido artigo. O SGPD monitora esse processamento, captura os parâmetros processados e cria a linhagem das imagens geradas durante o processamento. A linhagem em Python é então registrada em um arquivo XML seguindo o Esquema DC.

Durante o processamento algumas imagens intermediárias são geradas. Ao final do processamentos o SGPD gera a imagem resultante (Figura 9).

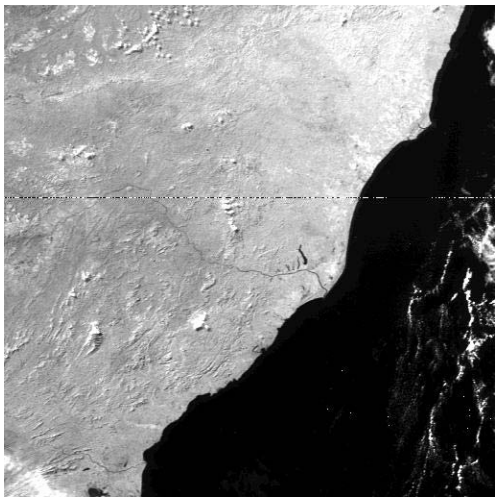

Figura 8 - Imagem NOAA original (com a presença de uma listra horizontal).

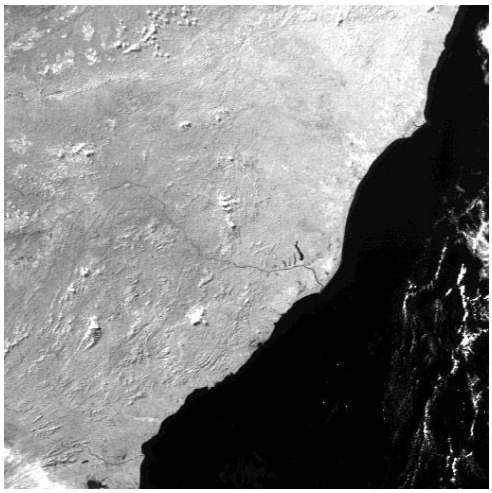

Figura 9 - Imagem resultante do processamento (sem o efeito de listras).

Através do SGPD podemos armazenar a imagem resultante e sua procedência na Biblioteca Digital UR*Lib*. Ao enviarmos os dados para a UR*Lib* o SGPD emite um formulário para preenchimento de informações relativas a origem da imagem resultante (Figura 10).

Observamos na Figura 10 que os campos solicitados são Título do processamento, Autores, Instituição, Descrição do processamento, E-mail, Palavras Chaves. Observamos também que as informações sobre o sistema de processamento utilizado é preenchido automaticamente pelo SGPD. As informações sobre origem e linhagem da imagem resultante são registradas no arquivo XML o qual descreve a procedência de imagens Figura 11.

Nome do Campo	Valor do Campo
Título	Exemplo Restauração de imagens NOAA por Morfologia Mate
Autor(es)	Banon, Gerald Jean Francis
Instituição	INPE
Descrição	Utilização de Morfologia Matemática para Reduzir o Efeito de Linha em imagens NOAA
E-Mail	banon@dpi.inpe.br
Palavras Chave	Imagens Noaa, Procedência, Linhagem
Sistema de Processamento	Gerado no SGPD 1.0 no Acervo URLib - dpi.inpe.br/juliana/2003/08.18.12.28 Linhagem em Python 2.2.: disponivel no Acervo URLib - dpi.inpe.br/juliana/2003/08.15.10.26

Figura 10 - Formulário para preenchimento da origem dos dados.

```
<Título> Exemplo Restauração de imagens NOAA por Morfologia Matemática </Título>
<author>Banon, Gerald Jean Francis</author>
<Palavras Chave> XML, Procedência </ Palavras Chave>
<Descrição> Redução do Efeito de Listras em imagens NOAA</Descrição>
<Formato>Gerado no SGPD 1.0 no Acervo URLib - dpi.inpe.br/juliana/2003/08.18.12.28
Linhagem em Python 2.2.: disponivel no Acervo URLib - dpi.inpe.br/juliana/2003/08.15.10.26 </Formato>
<Identificador> a1_dpi.inpe.br_juliana_2003_03_10_13_15 </Identificado>
<Idioma>Português</Idioma>
<Relação></Relação>
<Linhagem>import adpil
import morph
f1=adpil.adread('f02.gif')
f2=morph.mmclose(f1,morph.mmsesum(morph.mmimg2se
(morph.mmbinary([[0,0,0],[1,1,1],[0,0,0]])),30))
f2c=morph.mmero(f2,morph.mmimg2se(morph.mmbinary([[0,0,0],[0,0,0],[0,1,0]])))
f2cc=morph.mmcmp(f2c,'~=',f2)
f2b=morph.mmero(f2,morph.mmimg2se(morph.mmbinary([[0,1,0],[0,0,0],[0,0,0]])))
f2bb=morph.mmcmp(f2b,'~=',f2)
f2a=morph.uint8(morph.mmdil(f2,morph.mmimg2se(morph.mmbinary([[0,1,0],[0,1,0],[0,1,0]]))))
f2aa=morph.mmcmp(f2a,'==',f2)
f3=morph.mmintersec(f2aa,f2bb,f2cc)
f4a=morph.mmopen(f3,morph.mmsesum(morph.mmimg2se
(morph.mmbinary([[0,0,0],[1,1,1],[0,0,0]])),150))
f4b=f4a*255
f4=morph.uint8(f4b)
f5c=morph.mmero(f1,morph.mmimg2se(morph.mmbinary([[0,0,0],[0,1,0],[0,1,0]])))
f5b=morph.mmero(f1,morph.mmimg2se(morph.mmbinary([[0,1,0],[0,0,0],[0,1,0]])))
f5a=morph.mmero(f1,morph.mmimg2se(morph.mmbinary([[0,1,0],[0,1,0],[0,0,0]])))
f5=morph.mmunion(f5a,f5b,f5c)
f6= morph.mmunion(morph.mmintersec(f1,morph.mmneg(f4)),morph.mmintersec(f5,f4));
</Linhagem>
```

Figura 11 - Procedência da imagem resultante escrita em XML.

Para testar a reprodução de dados a partir de sua procedência, fechamos a sessão atual e abrimos uma nova sessão no SGPD. A nova sessão não contém nenhum dado processado.

Executamos o comando de busca para procurar a imagem resultante (Figura 9) na UR*Lib*. A busca é feita por título utilizando como palavra chave "NOAA" (Figura 12).

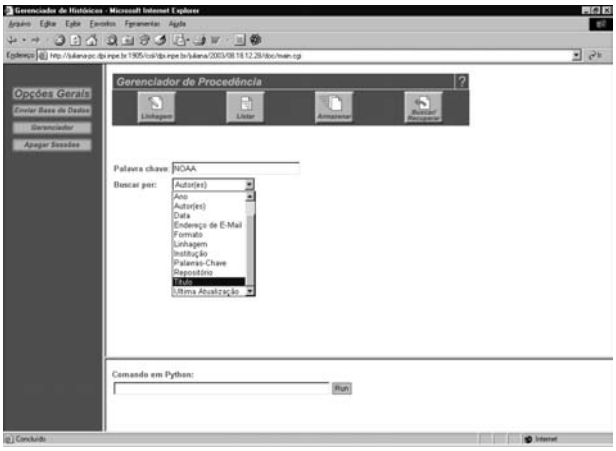

Figura 12 - Tela para realizar busca de dados na UR*Lib*.

A UR*Lib* retorna o resultado da busca (Figura 13).

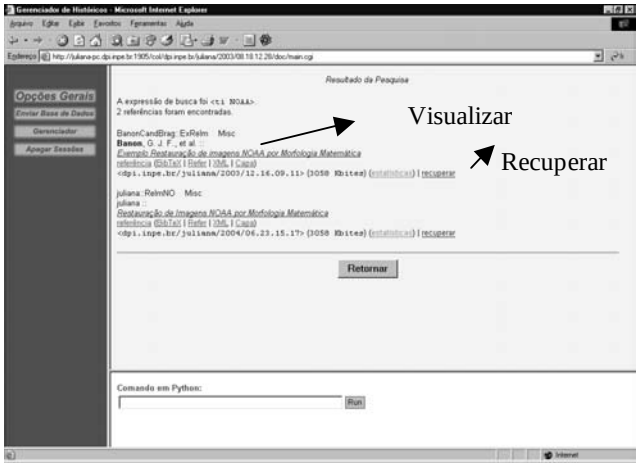

Figura 13 - Tela de retorno da busca realizada na UR*Lib*.

Através da tela de retorno da UR*Lib* podemos fazer o download da imagem resultante, consultar sua procedência ou reproduzir as imagens originais e intermediárias a partir da procedência da imagem resultante.

Para testar a reprodução de imagens a partir de sua procedência clicamos no link recuperar da tela de retorno da UR*Lib* (Figura 13).

O comando recupera as imagens originais, intermediárias e resultante para a sessão atual (Figura 14).

Observamos que as imagens intermediárias não foram guardadas na UR*Lib*, porém foram recuperadas a partir da procedência da imagem final.

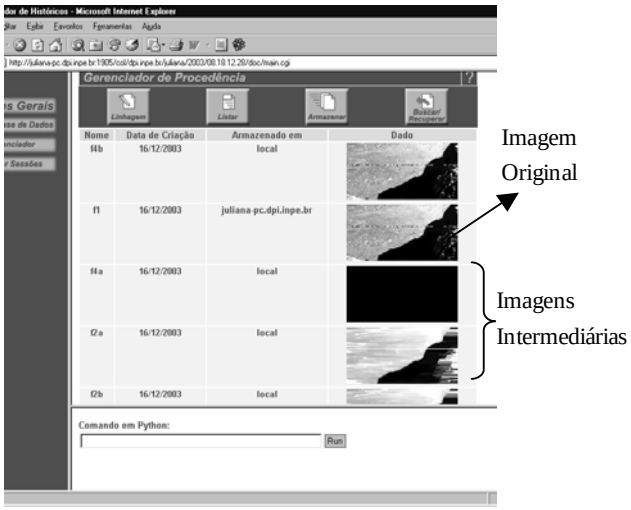

Figura 14 - Imagens recuperadas a partir de sua procedência.

5 Conclusões

Desenvolvemos neste trabalho a formalização da procedência de dados. A partir da formalização implementamos um protótipo para gerenciar a procedência. Testamos o protótipo para um exemplo em processamento de imagens.

O formalismo da procedência mostrou-se muito útil para orientar a implementação.

O armazenamento da procedência na UR*Lib* permitiu que processamentos realizados sejam compartilhados com toda a comunidade científica evitando uma possível repetição de esforços e facilitando a continuidade de trabalhos.

Pretende-se futuramente dar continuidade a esse trabalho nos seguintes aspectos:

- Permitir a alteração consistente da linhagem dos dados para posterior processamento.
- Registrar a linhagem em uma linguagem independente.
- Testar o uso do SGPD para outras aplicações além do processamento de imagem.
- Complementar a fração do metadado que define a origem de imagens de acordo com o tipo de aplicação (ex: imagens médicas, sensoriamento remoto, microscópios).
- Transformar o SGPD em um sistema colaborativo onde grupos de pesquisas em processamento de imagens possam compartilhar facilmente seus resultados
- Acrescentar ao modelo metadados que contenham informação a respeito do contexto das imagens.

6 Referências

As referências mencionando a URLib, podem ser encontradas a partir de qualquer site rodando o URLibService, por exemplo: <http://iris.sid.inpe.br:1905>. Basta introduzir o identificador (nome do repositório) no campo de busca.

[1] Banon, G, J, F. *Implementação de um sistema de tratamento de imagens usando uma definição ampla de imagens*; III Simpósio de Sensoriamento Remoto, Rio de Janeiro, RJ, 28-30 nov. 1991.

[2] Banon, G. J. F.; Banon, L. C. O que é a UR*Lib*? Disponível na biblioteca digital UR*Lib*: <iconet.com.br/banon/2001/05.25.16.44>. Acesso em: 25 fev. 2004.

[3] Banon, G. J. F.; Ribeiro, M.L.; Banon, L. C. Preservação digital da memória técnico-científica do INPE. II Simpósio Internacional de Biblioteca Digital. Disponível na biblioteca digital UR*Lib*: <dpi.inpe.br/lise/2004/03.02.15.20>. Acesso em: 8 fev. 2004.

[4] Banon, G. J. F.; Candeias, A. L. B. Restauração de imagens NOAA por morfologia matemática. In: SIBGRAPI 93., VI Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens, October, Wagner Ltda., 1993. p. 139-145.

[5] Barrera, J.; Banon, G. J. F.; Lotufo, R. A.; Hirata Junior, R. MMach: a mathematical morphology toolbox for the KHOROS system. J. of Electronic Imaging, v. 7, n. 1, p. 174-210, Jan. 1998.

[6] Buneman, P.; Khanna, S.; Tan, W. Why and Where: A Characterization of Data Provenance. In: International Conference on Database Theory, 4-6 January, London, United Kingdom. Proceedings... 2001

[7] Especificação XML; eXtensible Markup Language. <http://www.w3.org/TR/XML>. abr. 2004.

[8] Goble, C. Position Statement: Musings on Provenance, Workflow and (Semantic Web) Annotations for Bioinformatics. In: Workshop on Data Derivation and Provenance, 17-18 Oct., Chicago, Illinois. Proceedings... 2002.

[9] Harrison, A M.; Introduction to Formal Language Theory. : Addison-Wesley Publishing Company, Inc, 1978. 594 p.

[10] Página Oficial do Python: <http://www.python.org>

[11] Woodruff, A.; Stonebraker, M. Supporting Fine-Grained Data Lineage in a Database Visualization. In: 13th International Conference on Data Engineering, April 7-11, Birmingham, UK. Proceedings... 1997. p. 15.