

FACULDADES INTEGRADAS DE CARATINGA

Ciência da Computação

DESENVOLVIMENTO DE UMA INFRAESTRUTURA PARA
CATALOGAÇÃO DE PADRÕES DE ANÁLISE

Joabe Machado de Abreu

CARATINGA

2013

Joabe Machado de Abreu

**DESENVOLVIMENTO DE UMA INFRAESTRUTURA PARA CATALOGAÇÃO DE
PADRÕES DE ANÁLISE**

Monografia apresentada à Faculdade de
Ciência da Computação das Faculdades
Integradas de Caratinga como exigência
parcial da disciplina de Trabalho de
Conclusão de Curso sob orientação do
Professor Msc. Glauber Luis da Silva
Costa.

CARATINGA

2013

Joabe Machado de Abreu

DESENVOLVIMENTO DE UMA INFRAESTRUTURA PARA CATALOGAÇÃO
DE PADRÕES DE ANÁLISE

Monografia submetida à Comissão examinadora designada pelo Curso de Graduação em Ciência da Computação das Faculdades Integradas de Caratinga como requisito para obtenção do grau de Bacharel.

Prof. Msc. Glauber Costa

Faculdades Integradas de Caratinga

Prof. Msc. Fabrícia Pires

Faculdades Integradas de Caratinga

Prof. Msc. Paulo Eustáquio dos Santos

Faculdades Integradas de Caratinga

Caratinga, 09/12/2013

DEDICATÓRIA

Dedico este trabalho, primeiramente a Jesus Cristo, Filho de Deus, autor e consumidor da minha fé, e à minha mãe, Eva Maria Machado, que sonhou juntamente comigo, com esta vitória.

AGRADECIMENTOS

Agradeço, primeiramente, a Deus que me proveu tudo que foi necessário para alcançar essa vitória, a minha mãe, Eva Maria Machado, que foi meu apoio financeiro e me ensinou a nunca desistir em meios aos imprevistos que aparecem na vida, e as pessoas que Deus levantou para me ajudar nesse caminho, algumas dessas pessoas estão listadas abaixo:

- Luciano Magela Campos, em 2010, diretor do Departamento de Juventude de Caratinga, que me colocou em contato com o diretor da Rede de Ensino Doctum;
- Dr. Pedro Leitão, vice-presidente da Rede de Ensino Doctum, que autorizou a Msc. Fabrícia Pires a me dar uma vaga de emprego na instituição, emprego que me garantiu desconto nas mensalidades que, sem elas, não conseguiria chegar até aqui;
- Msc. Fabrícia Pires, que me arrumou uma vaga de emprego na Rede Doctum, mediante o pedido do vice-presidente da Rede, e que, dentre outras coisas, me aconselhou durante todo curso nas diversas situações que ocorreram;
- Msc. Glauber Luis da Silva Costa, que me ofereceu o tema deste trabalho, me proveu parte do referencial teórico utilizado no mesmo e me orientou neste trabalho.
- Bruno Rocha Spínola, que foi meu companheiro desde o início dessa graduação, sempre me ajudando nas disciplinas em que encontrei dificuldades.

A todas essas pessoas que Deus levantou para me ajudar, e a outras que não tem como citar aqui, meu muito obrigado.

EPÍGRAFE

Ao único Deus, sábio, Salvador nosso, por Jesus Cristo, nosso Senhor, seja glória e majestade, domínio e poder, antes de todos os séculos, agora, e para todo o sempre. Amém! Judas 1.25

RESUMO

Padrões de análise colaboram muito no desenvolvimento de software, provendo artefatos computacionais comprovadamente úteis, que podem ser reutilizados, evitando retrabalho e acelerando assim o processo de desenvolvimento de software.

Mesmo sendo um recurso útil, o acesso à padrões de análise é muito deficiente por diversos fatores, como por exemplo, o custo que se tem para comprar um livro que contém o padrão de análise desejado, e para solucionar esse problema, foi proposta uma infraestrutura que soluciona os principais problemas no que se refere ao reuso de padrões de análise, essa infraestrutura foi intitulada como *Analysis Patterns Reuse Infrastructure*.

Apesar de não se ter conhecimento de algum software que possa prover todas as funcionalidades da desta infraestrutura, sabe-se que existem diversas ferramentas que, se corretamente integradas e configuradas, podem realizar as atividades propostas, isto é, gerar os mesmos resultados esperados.

Esse trabalho propõe integrar algumas ferramentas e customizá-las de modo que viabilizem tecnologicamente a *Analysis Patterns Reuse Infrastructure*, de modo que padrões de análise possam ser documentados, armazenados e distribuídos de forma online, para que os usuários (desenvolvedores, arquitetos de software, pesquisadores, etc.) consigam usufruir desses artefatos computacionais para agilizar qualquer trabalho que dependa desses artefatos.

Palavras-chave: Padrões, Reusabilidade, Padrões de Análises, Metadados, Portal.

ABSTRACT

Analysis patterns collaborate much in software development, providing computational artifacts proven useful, which can be reused, avoiding rework and thus accelerating the software development process.

Even being an useful resource, the access to analysis patterns is very deficient by many factors and, to solve this problem, was proposed a infrastructure that solves the main problems regarding the reuse of analysis patterns. This infrastructure was entitled as Analysis Patterns Reuse Infrastructure.

Although there is no known software which can provide all of the functionalities of this infrastructure, it is known that exists diverse tools that, if integrated and configured correctly, can realize the proposed activities, that is, generate the same expected results.

This work proposes to integrate some tools and, if needed, customize them so that they make technologically possible the Analysis Patterns Reuse Infrastructure, so that analysis patterns can be documented, stored and distributed in an online form, so that the users (developers, software architects, researchers etc.) may utilize these computational artifacts to accelerate any work that relies on these artifacts.

Keywords: Patterns, Reusability, Analysis Patterns, Metadata, Portal.

LISTA DE FIGURAS

FIGURA 1 – Exemplo de código XML	18
FIGURA 2 – Funcionamento dos <i>Web Services</i>	20
FIGURA 3 – Serviço descrito em WSDL	22
FIGURA 4 – Padrão de análise <i>Party</i>	26
FIGURA 5 – Projeto da APRI.....	27
FIGURA 6 – Tela inicial do Fedora	29
FIGURA 7 – Tela inicial do Drupal	31
FIGURA 8 – Tela inicial do GSearch	34
FIGURA 9 – Tela Inicial do Solr	35
FIGURA 10 – Padrão de análise sendo documentado	44
FIGURA 11 – Exibição do padrão de análise <i>Organization Hierarchies</i>	45
FIGURA 12 – Local do serviço de busca por padrões de análise	46

LISTA DE SIGLAS

APRI – *Analysis Patterns Reuse Infrastructure*

CMS – *Content Management System*

CModel – *Content Model*

CSS – *Cascading Styles Sheets*

DC – *Dublin Core*

DC2AP – *Dublin Core Application Profile to Analysis Patterns*

Fedora - *Flexible Extensible Digital Object Repository Architecture*

FDO – *Fedora Digital Object*

GSearch – *Generic Search*

HTML - *HyperText Markup Language*

HTTP - *Hypertext Transfer Protocol*

SOAP - *Simple Object Access Protocol*

SP – *Solution Pack*

WSDL - *Web Services Description Language*

XML - *Extensible Markup Language*

SUMÁRIO

1. INTRODUÇÃO.....	13
2. REFERÊNCIAL TEÓRICO	15
2.1. CONCEITOS IMPORTANTES	15
2.1.1. Reusabilidade	16
2.1.2. <i>Extensible Markup Language</i>	18
2.1.3. <i>Web Services</i>	19
2.1.4. <i>Simple Object Access Protocol</i>	21
2.1.5. WSDL	21
2.1.6. Metadados	22
2.1.7. <i>Dublin Core Metadata Initiative</i>	23
2.1.8. <i>Content management system</i>	24
2.2. PADRÕES DE ANÁLISE.....	24
2.2.1. O uso de padrões	25
2.2.2. Padrões de análise	25
2.3. <i>ANALYSIS PATTERNS REUSE INFRAESTRUCTURE</i>	26
2.4. <i>Dublin Core Application Profile to Analysis Patterns</i>	28
2.5. FERRAMENTAS	28
2.5.1. <i>Flexible Extensible Digital Object Repository Architecture</i>	28
2.5.2. Drupal	30
2.5.3. <i>Islandora Framework</i>	32
2.5.4. <i>Fedora Generic Search</i>	33
2.5.5. Apache Solr	34
3. METODOLOGIA	36
3.1. O CONTEXTO ONDE O TRABALHO FOI REALIZADO	36
3.2. OBTENÇÃO DAS FERRAMENTAS.....	37
3.3. INSTALAÇÃO	37
3.4. CUSTOMIZAÇÕES DAS FERRAMENTAS.....	38
3.4.1. Drupal	38
3.4.2. GSearch	40
3.4.3. Solr	40
3.5. Criação do Pandora SP.....	41
3.6. Testes	41

4. ANÁLISE DOS RESULTADOS.....	43
4.1. Inserção dos padrões de análises.....	43
4.2. Exibição dos padrões de análises inseridos.....	44
4.3. Busca pelo padrão de análise <i>Organization Hierarchies</i>	45
5. Conclusão.....	47
6. Trabalho futuro	48
7. REFERÊNCIAS	49
Apêndice 1 – Alteração feita na função <code>fedora_repository_object_details_table</code> do arquivo <code>ObjectDetails</code> alterada	54
Apêndice 2 – Funções adicionadas no arquivo <code>ObjectDetails.inc</code>	55
Apêndice 3 – Alteração feita na função <code>theme</code> , localizada no arquivo <code>theme.inc</code>	56
Apêndice 4 - Funções adicionadas no arquivo <code>theme.inc</code>	57
Apêndice 5 – Modelo de conteúdo criado para o Pandora SP (arquivo <code>STRICT_SP_PANDORACM.xml</code>).....	62
Apêndice 6 – Arquivo que define a qual modelo de conteúdo deverá ser usado pela coleção Pandora (arquivo <code>PANDORA - COLLECTION POLICY.xml</code>).....	63
Apêndice 7 – Alteração feita no arquivo <code>schema.xml</code> para indexar metadados Pandora	64
Apêndice 8 – Alteração feita no arquivo <code>solrconfig.xml</code> para que o sistema de busca funcione	65
Apêndice 9 – Arquivo <code>foxToSolr.xslt</code> (responsável por recuperar os metadados DC2AP e formata-los para serem indexados pelo Solr	66

1. INTRODUÇÃO

O processo de desenvolvimento de um software é composto de diversas partes, em cada uma delas procura-se empregar a reutilização. Uma das maneiras de se empregar a reutilização é a utilização de padrões. Existem diversos padrões (padrões de análise, padrões de projeto, padrões de arquitetura, padrões de implementação, etc.) e é importante recorrer a eles sempre que possível, pois economizam tempo e dinheiro, independente do que estiver sendo desenvolvido, além de incrementar a qualidade dos destes produtos.

Os padrões de análise são artefatos voltados para a reutilização durante as etapas de análise de requisitos e modelagem conceitual (Vegi, 2012), contudo os padrões de análises são pouco conhecidos, pois as formas com que são divulgados não proporcionam uma ampla distribuição dos mesmos, fazendo com que esse recurso seja pouco utilizado.

A pouca, ou nenhuma, utilização dos padrões de análise, traz o problema de realizar um trabalho que já foi realizado por alguém, com isso, torna-se necessário gastar um tempo que poderia ser aproveitado para realização de outras tarefas e, conseqüentemente, isto irá gerar custo. Como um padrão nunca cobre todas as situações e necessita de revisões para que ocorra uma constante melhoria, no que se refere aos seus detalhes, pesquisadores também são prejudicados com a ausência dos padrões de análise, visto que todo o trabalho deve ser feito do zero, sendo que, se o trabalho partisse de um padrão já existente, a melhoria seria feita em menos tempo e, no final, ter-se-ia um padrão mais rico em detalhes, mais abrangente.

Para solucionar esse problema, foi proposto por Vegi, et al. (2012) uma infraestrutura para catalogação de padrões de análise, conhecida como *Analysis Patterns Reuse Infrastructure* (APRI), nos quais é possível documentar padrões de análises e armazená-los para acesso dos usuários, essa infraestrutura leva em consideração um conjunto das principais necessidades no que se refere a documentação e acesso de padrões de análises.

Apesar de não se ter conhecimento, de algum software que possa prover todas as funcionalidades da APRI, sabe-se que existem diversas ferramentas disponíveis que, se corretamente interligadas e configuradas, podem prover todas as funcionalidades da APRI, sendo as principais são:

- Fedora *Repository*;
- Islandora *Framework*;
- Drupal;

A solução proposta neste trabalho consiste em combinar estas ferramentas, de modo que elas possam prover as funcionalidades que a APRI se propõe, para que os padrões de análise, possam ser facilmente documentados, armazenados, divulgados e utilizados por qualquer profissional que necessite desse recurso para agilizar, ou melhorar, seu trabalho.

Para se alcançar o objetivo deste trabalho, diversas ferramentas serão instaladas, integradas e customizadas.

2. REFERÊNCIAL TÉORICO

Este capítulo tem por objetivo proporcionar condições necessárias para o entendimento da solução proposta neste trabalho, como também de assuntos relacionados a solução que será apresentada.

Na seção 2.1 será tratado assuntos relacionados com a solução proposta. Na seção 2.2 trataremos sobre padrões de análises. Na seção 2.3 será tratada a APRI, a infraestrutura que objetiva o armazenamento digital, confiável e uma fácil disseminação de padrões de análises. Na seção 2.4 será tratado o perfil de metadados criado para se documentar padrões de análise, o *Dublin Core to Analysis Patterns* (DC2AP). Na seção 2.5 serão apresentadas as ferramentas que serão necessárias para o desenvolvimento da APRI.

2.1. CONCEITOS IMPORTANTES

A solução que será apresentada neste trabalho é composta por um conjunto de tecnologias, que recentemente tem se tornado o alvo de desenvolvedores de softwares. Tecnologias como *Web services*, indexação de arquivos e gerenciadores de conteúdo, fazem parte da essência da solução proposta, seja para armazenar conteúdo, recuperá-los ou exibidos para o usuário final.

O mal entendimento desses, e outros, conceitos podem inviabilizar a correta abstração da viabilização tecnológica da APRI, por esse motivo, essa seção tem o objetivo de explanar todos os conceitos que são necessários para o perfeito entendimento da solução proposta neste trabalho.

2.1.1. Reusabilidade

Em 1968 ocorreu a Conferência de Engenharia de Software NATO, nos quais o foco dessa conferência eram os diversos problemas que a fabricação de software estava enfrentando (surgindo então o termo crise do software). Nesta conferência, foi apresentado um periódico escrito por McIlroy, onde foi proposta de uma biblioteca de componentes reutilizáveis e técnicas automatizadas para customizações de componentes em diferentes níveis de precisão e robustez (Krueger, 1992), a partir dessa conferência passou a existir na engenharia de software a reusabilidade.

A reusabilidade tem como objetivos: qualidade, produtividade e a efetividade na fabricação e manutenção de softwares. Uma vez que partes criadas anteriormente e já testadas de um sistema forem reutilizadas, resta apenas o trabalho de adaptar as partes reutilizáveis ao novo sistema (Resende, 2005).

Embora a reutilização tenha sido considerada a chave para solucionar os problemas concernentes a crise do software, a realidade mostrou que não seria tão fácil criar artefatos que realmente atendessem todas as características de um artefato reutilizável (Lucrédio, 2009), dentre essas características, Krueger (1992) cita:

- **Abstração:** A principal característica da reutilização. Consiste em uma descrição sucinta das informações mais importantes do artefato, onde o desenvolvedor pode, sem analisar o artefato, saber se ele atende suas necessidades, se precisa fazer alguma adaptação, ou se o artefato é passível ou não de ser utilizado em determinado contexto (Lucrédio, 2009);
- **Seleção:** Consiste em alguma técnica para que o desenvolvedor encontre o mais rápido possível, em meio a uma biblioteca de artefatos, algum que lhe seja útil;
- **Adaptação:** Para diminuir o esforço gasto para adaptar artefatos a um novo contexto, procura-se criar artefatos genéricos, que atendem as mais diversas aplicações possíveis; e
- **Integração:** Consiste em tentar manter a consistência e padronização das interfaces, prever as possíveis necessidades de adaptação ou modificação, e realizar testes de forma a validar a funcionalidade em nível de sistema (Lucrédio, 2009).

Segundo Sommerville (2011) apesar da ideia de aplicar o reuso no desenvolvimento de softwares ter sido proposta há mais de 40 anos somente a partir de 2000 ela se tornou norma para novos sistemas empresariais. Com a modernização da produção de software e o surgimento de técnicas de criação de softwares, o reuso tornou-se preocupação dos profissionais ligados ao desenvolvimento de softwares. Entretanto só muito recentemente os engenheiros de software puderam aplicar o reuso de forma abrangente (Braude, 2005).

Devido a tudo que o software representa nos dias de hoje, reduzir os custos de um software significa um lucro maior para o fabricante e economia para o cliente. Por isso, investimentos destinados a reduzir prazos e custos na produção de software são altamente desejáveis (Paula, 2000) e, para solucionar esse problema, o reuso aparece como uma importante ferramenta. Peters (2001) diz que o reuso proporciona, por exemplo, a possibilidade de economizar tempo na resolução de um problema de manutenção, além disso, diversos benefícios são citados por Sommerville (2011) a respeito da reutilização de softwares, são eles:

- **Confiabilidade:** Artefatos reutilizáveis são testados e provados, isto faz com que se tornem mais confiáveis que novos softwares;
- **Risco de processos reduzidos:** Redução na margem de erro na estimativa de custo do projeto;
- **Uso efetivo de especialistas:** Não há a necessidade de fazer o mesmo trabalho várias vezes, por exemplo, um engenheiro de software pode recorrer à padrões de análise para acelerar o planejamento do software.
- **Cumprimento de padrões:** Alguns padrões, tais como padrões de interface, podem ser implementados como um conjunto de componentes reutilizáveis.
- **Desenvolvimento acelerado:** Ao se utilizar partes que já foram criadas e testadas, o desenvolvimento é mais rápido, fazendo com que o tempo de desenvolvimento seja o menos possível.

Por todas esses benefícios conclui-se que a reusabilidade se apresenta como uma solução satisfatória para os diversos problemas relacionados a softwares nos dias atuais.

Mediante tudo que o reuso representa para as empresas e para os próprios softwares, todos os componentes de software podem ser reutilizáveis. É comum que sistemas, mesmo que criados com finalidades diferentes, possuam pontos em comum (REZENDE, 2005).

2.1.2. *Extensible Markup Language*

A *Extensible Markup Language* (XML) é um simples arquivo de texto que representam informações estruturadas que surgiu a partir de um pequeno grupo de trabalho, realizado em meados da década de 90, com o objetivo de simplificar a SGML (uma metalinguagem existente na época) (R. E. Quin, n.d.).

O XML é usado, principalmente, em duas situações: A primeira é representar dados baixo-nível, como arquivos de configuração e a segunda é adicionar metadados em documentos. Abaixo, na FIGURA 1, tem-se um exemplo de código XML (Fawcett, Ayers, & Liam R.E., 2012):

```
<lembrete>  
<para>Marcela</para>  
<de>Marcos </de>  
<titulo>Lembrete !</titulo>  
<mensagem>Não se esqueça do horário </mensagem>  
</lembrete>
```

FIGURA 1 – Exemplo de código XML
Fonte: Próprio Autor

Na FIGURA 1 vê-se um código XML, as marcações aparecem entre parêntesis angulares (<>) e os dados que estão sendo marcados aparecem realçados em negrito.

O primeiro trabalho em XML foi publicado em 1996 e dois anos mais tarde passou a ser recomendada pela *World Wide Web Consortium (W3C)*, organização que tem como objetivo padronizar a *World Wide Web* (Fawcett et al., 2012).

Quem está familiarizado com a linguagem *HyperText Markup Language* (HTML) notará semelhanças entre esta e o XML, contudo existem diversas diferenças entre elas já que são linguagens com objetivos diferentes, o HTML tem o objetivo de marcar como determinado conteúdo deve ser exibido, já o XML tem o objetivo de descrever o conteúdo que está sendo marcado, e por terem objetivos diferentes, surgem então algumas diferenças, como as citadas por R. E. Quin:

- Todos os elementos em XML precisam ser fechados ou marcados como vazios;
- Em XML, os valores dos atributos precisam, obrigatoriamente, de estarem entre "" (aspas duplas);
- XML possui um conjunto de nomes de elementos predefinidos;
- XML trabalha com caracteres Unicode;
- XML permite utilizar valores hexadecimais em referências de caracteres numéricos.

Por essas, e outras características, o XML é amplamente utilizado nos dias atuais, como, por exemplo, em *Web services*, portanto, conhecer essa linguagem é crucial para se entender diversas tecnologias presentes nos nossos dias.

2.1.3. Web Services

A *Web* iniciou suportando interações humanas com dados textuais e imagens. As pessoas passaram a utilizar a *Web* para os mais diversos fins, como comprar, vender, transações bancárias e ler as últimas notícias. Para estes propósitos a *Web* funciona muito bem, contudo, quando se trata de um software interagindo com a *Web*, dados textuais não funcionam muito bem, principalmente quando se trata grandes transferências de informações (Newcomer, 2004).

Para suprir essa necessidade, surgiu a proposta dos *Web Services*, que é qualquer serviço disponível na *web* usando um sistema padronizado de troca de mensagens XML, e está desacoplado de qualquer linguagem de programação

(Cerami, 2002). Através desse sistema de troca de mensagens, os *Web Services* definem um padrão para o formato da mensagem, especifica uma interface para o envio das mensagens, descreve convenções para mapear o conteúdo da mensagem, do programa que implementa o serviço, e define mecanismos para publicar e para encontrar interfaces *Web services* (Newcomer, 2004).

Para que essa troca de mensagens XML seja feita com sucesso os *Web services* trabalham sobre algum protocolo, onde o principal deles é o protocolo *Simple Object Access Protocol* (SOAP), e são descritos em *Web Services Description Language* (WSDL). Através da FIGURA 2 pode-se ter uma visão panorâmica de como funcionam os *web services*.

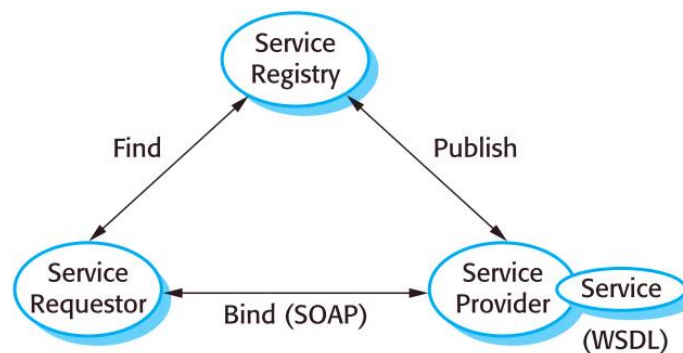


FIGURA 2 – Funcionamento dos *Web Services*
Fonte: *Software Engineering* 9th ed.

Sommerville (2011), através da FIGURA 2, mostra que o *Service Requestor* (solicitador de serviço, também chamado de cliente de serviço) procura pelo registro do serviço desejado em *Service Registry* e, através do protocolo SOAP, faz o pedido do mesmo para o *Service Provider* (provedor de serviços) que contém o serviços (*Service*) descritos em WSDL.

Entender o funcionamento dos *Web services* é importante para uma melhor compreensão do funcionamento da solução proposta neste trabalho.

Como os *web services* trabalham com troca de mensagens XML é necessário um básico conhecimento sobre a linguagem XML para uma boa abstração desse conceito.

2.1.4. Simple Object Access Protocol

O protocolo SOAP define uma forma padronizada para troca de mensagens XML sobre o protocolo *Hypertext Transfer Protocol* (HTTP).

SOAP é, fundamentalmente, um modelo de comunicação que garante a coerência das mensagens que estão sendo enviadas (Newcomer, 2004), ele define os componentes essenciais e opcionais das mensagens que estão sendo enviadas entre os serviços (Sommerville, 2011).

A especificação do SOAP o define como um *framework*, simples, completamente neutro no que se refere a sistemas operacionais, linguagem de programação ou plataforma, para troca de dados em formato XML.

No contexto de *Web Services* o XML não é usado apenas como um formato de mensagem, mas também para descrever os serviços (Newcomer, 2004).

Entender o protocolo SOAP é importante para entendermos o funcionamento dos *Web services*, já que é através dele que ocorre o transporte de dados.

2.1.5. WSDL

A WSDL é um padrão, ou um esquema XML, que define um *framework* para descrever uma interface de um serviço (Newcomer, 2004) (Sommerville, 2011).

O WSDL é a parte principal dos *Web Services*, com o WSDL é possível descrever uma forma padrão de comunicação para representar tipos de dados nas mensagens, as operações a serem realizadas, e mapear as mensagens no transporte pela rede (Newcomer, 2004). Segundo Cerami (2012), WSDL descreve quatro peças-chaves no que se refere a *Web Services*:

- Descrição de todas as funções públicas disponíveis;
- Tipos de dados para todas as solicitações e respostas de mensagens;
- Informações sobre o tipo de protocolo que está sendo utilizado; e
- A localização do serviço específico.

Na FIGURA 3 vê-se um exemplo de um serviço descrito em WSDL.

```
▼<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  ▼<soap:Body>
    ▼<soap:Fault>
      <faultcode>soap:Server</faultcode>
      ▼<faultstring>
        No such operation: (HTTP GET PATH_INFO: /fedora/services/access)
      </faultstring>
    </soap:Fault>
  </soap:Body>
</soap:Envelope>
```

FIGURA 3 – Serviço descrito em WSDL
Fonte: Próprio Autor

2.1.6. Metadados

Metadados são dados sobre dados, ou seja, informações que descrevem outras informações (Vegi, 2012), informando, por exemplo, se determinada informação é o título, se é a descrição ou se é o assunto do documento.

Os metadados tem por principal objetivo descrever, identificar e definir um recurso, o que os torna importantes para gestão, organização e recuperação de informações digitais.

Os metadados são cruciais para que haja interoperabilidade entre aplicações, pois sua utilização permite que a informação que está sendo descrita seja compreendida não somente por pessoas, mas também por programas (Alves & Souza, 2007).

2.1.6.1. Padrões de Metadados

Um padrão de metadados é uma estrutura usada para descrever dados. O foco de um padrão de metadados está em um domínio de dados, assim, cada padrão de metadados, disponibiliza metadados referentes àquele domínio que se propõe a descrever, atendendo assim, peculiaridades específicas daquele domínio (Vegi, 2012), a exemplo disso tem-se o *Content Standard for Digital Geospatial Metadata* que é um padrão de metadados muito usado para descrever informações espaciais (Vegi, Peixoto, Lisboa Filho, & Oliveira, 2012b).

2.1.7. Dublin Core Metadata Initiative

Dublin Core (DC) é um padrão de metadados proposto em meados da década 90 a partir de ideia de um corpo de pesquisadores de diversas áreas que se reuniram na cidade de Dublin, no estado americano Ohio, com a intenção de obter os principais elementos que fossem capazes de descrever recursos de diferentes áreas do conhecimento humano (DCMI, n.d.).

A reunião culminou em um acordo de 15 elementos conhecidos como *Simple Dublin Core* (que é composto por 15 elementos originais), e também como *Qualified Dublin Core* (composto por mais 4 metadados além dos originais) (Vegi, 2012). Os quinze elementos originais são:

- *Title*
- *Creator*
- *Subject*
- *Description*
- *Publisher*
- *Contributor*
- *Date*
- *Type*
- *Format*
- *Identifier*
- *Source*
- *Language*

- *Relation*
- *Coverage*
- *Rights*

Nenhum dos metadados DC é obrigatório e pode ser repetido quantas vezes for necessário, durante a descrição das informações, gerando grandes benefícios na interoperabilidade entre dados de domínios diferentes (Vegi et al., 2012).

2.1.8. Content management system

Um Content Management system (CMS) é um aplicativo que ajuda na criação da estrutura de um *Web site*, como também no gerenciamento de todo conteúdo (Hauschildt, 2010).

Para Fraser (2002), um CMS gerencia os componentes de conteúdo de um site, assumindo como conteúdo o conjunto formado por todas as informações de um *Web site*, essas informações podem ser exibidas de duas formas:

- Como textos e imagens; e,
- Como partes de um *Web site* (como uma seção ou um menu).

O principal objetivo de um CMS é a separação da conteúdo, projeto, e programação, dessa forma, não é necessário ter qualquer conhecimento de linguagens de programação para publicar conteúdo, pois essa camada é abstraída pelo CMS (Hauschildt, 2010).

2.2. PADRÕES DE ANÁLISE

Para que um haja um bom desenvolvimento nas etapas de desenvolvimento de software (requerimentos, análise, design e desenvolvimento), o reuso tem de ser aplicado desde o início, e a utilização de padrões é considerada uma excelente

técnica de reuso (PANTOQUILHO *et al.* 2003).

2.2.1. O uso de padrões

Uma das maneiras de se empregar a reutilização é a utilização de padrões GAMMA *et al.* (1994), que podem ser entendidos como:

“Uma forma original plenamente realizada; modelo proposto ou aceito por imitação. Com padrões, pequenas partes são padronizadas em grandes partes ou unidades. Padrões se tornam blocos de construção para projetos e construções” (Coad, 1992).

Segundo GAMMA (2002), cada padrão descreve um problema no ambiente e o núcleo da sua solução, de tal forma pode-se usar esta solução mais de um milhão de vezes, sem nunca fazê-lo da mesma maneira.

Existem diversos tipos de padrões (como padrões de arquitetura, padrões de implementação) (Blaimer, Bortfeldt, & Pankratz, 2010) e, dentre eles, existem os padrões de análise.

2.2.2. Padrões de análise

Apesar desse conceito existir há um bom tempo, devido a diversas limitações, a ideia de padrões de análises é pouco conhecida. A disseminação dos padrões de análise encontra diversos desafios como, por exemplo, a disseminação em livros, onde o acesso é limitado em diversos sentidos.

Padrões de análise tem por objetivo ajudar a entender o problema em si (Hamza, 2002), através deles pode-se evitar cometer erros, uma vez que, os padrões de análise apresentam um modelo de ideias úteis a ser seguido (Pantoquilha, Raminhos, & Araújo, 2003), proporcionando assim resultados mais consistentes, com menor custo possível.

Fowler (1996) propõe vários exemplos de padrões de análise, entre esses exemplos tem-se um padrão feito a partir da análise de um catálogo de endereços. Fowler observou que em todo o catálogo havia endereços, telefones e *e-mails*, e a partir dessa análise, ele criou o padrão de análise *Party*, que é mostrado na FIGURA 4.

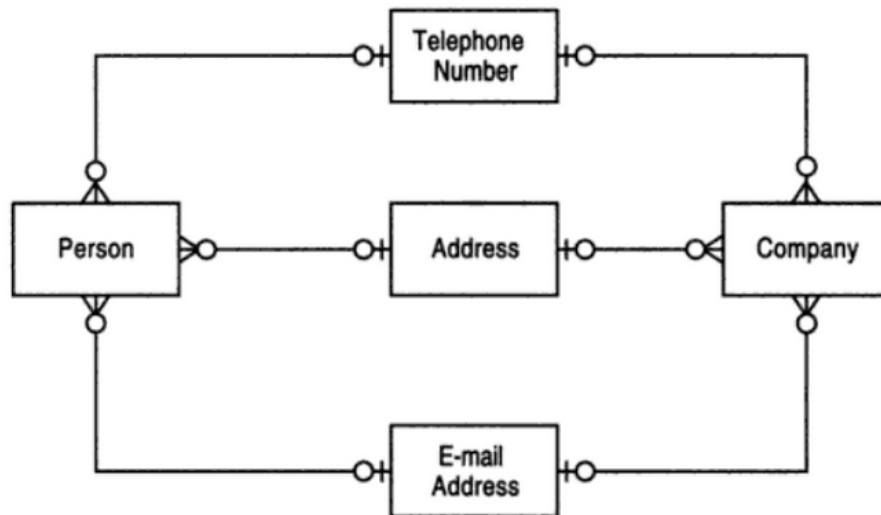


FIGURA 4 – Padrão de análise *Party*
Fonte: *Analysis Patterns: Reusable Object Models*

Na FIGURA 4, pode-se ver os três elementos observados por Fowler (1996) e vê-se ainda que esses três elementos são usados por *Person* e *Company* em um relacionamento 1:N, ou seja, tanto *Person* como *Company* podem ter vários telefones, endereços e *e-mails*.

Segundo Fowler (1996), esse é um típico caso de um conceito sem nome, isto é, algo que todos sabem e usam mas, ninguém dá nome.

2.3. ANALYSIS PATTERNS REUSE INFRASTRUCTURE

Mediante diversos problemas relacionados à disseminação dos padrões de análise, o grupo de Sistemas de Informação da Universidade Federal de Viçosa propôs a criação de uma Infraestrutura que possibilitasse armazenar e divulgar os padrões de análise de forma digital e Inter operável, para que não somente pessoas pudessem ter acesso aos padrões de análise mas também outras máquinas.

A infraestrutura proposta pelo grupo foi intitulada como *Analysis Patterns Reuse Infrastructure* (APRI) (Vegi et al., 2012b). A APRI, compreende *Web Services* e um perfil de metadados para a especificação de padrões de análise conhecido como DC2AP (FIGURA 5).

O objetivo da APRI é reduzir as deficiências no que se refere a padrões de análises, apoiar a catalogação e incentivar a reutilização dos padrões de análise (Vegi et al., 2012b).

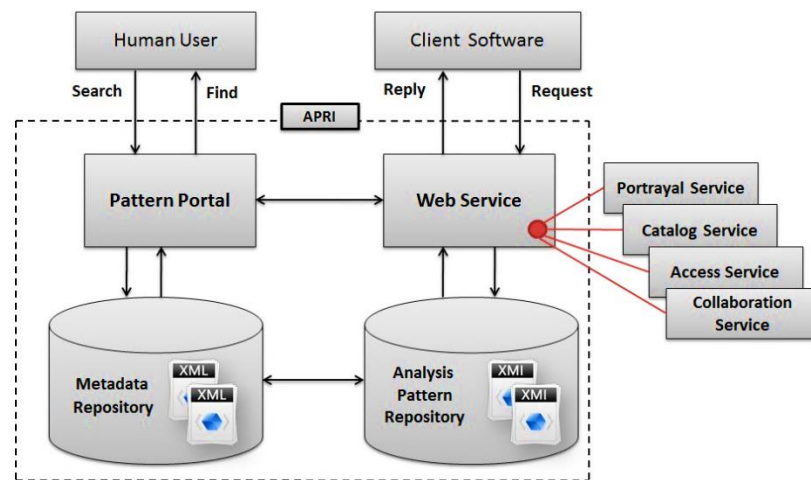


FIGURA 5 – Projeto da APRI

Fonte: APRI

Como pode-se observar na FIGURA 5, a APRI é composta basicamente por 4 (quatro) partes:

- *Pattern Portal* (Portal de Padrões): Onde os usuários (*Human User*) poderão acessar todas as documentações de padrões de análise do repositório.
- *Web Service* (Serviços Web): Serão utilizados pelo portal ou por qualquer outro programa (*Client Software*) que acesse Web Services;
- *Metadata Repository* (Repositório de Metadados): Onde os metadados ficarão armazenados;
- *Analysis Pattern Repository* (Repositório de Padrões de Análise): Lugar onde todos os padrões de análise documentados, segundo o perfil armazenado no Repositório de Metadados, serão guardados e poderão ser acessados pelos usuários pelo Portal de Padrões, via Serviços Web.

2.4. Dublin Core Application Profile to Analysis Patterns

Tendo como base a proposta da APRI, Vegi (2012) apresenta um perfil de metadados específico para a documentação de padrões de análise, intitulado como DC2AP.

O objetivo do DC2AP é criar uma forma padrão e completa para se catalogar padrões de análise, para isso fez um mapeamento entre metadados *Dublin Core* e um modelo proposto por Pantoquilha et al. (2003)

Algo importante a ser destacado é que, neste mapeamento, houve um refinamento entre os metadados existentes no *Dublin Core* e os metadados existentes no modelo proposto por (Pantoquilha et al., 2003), neste refinamento, as redundâncias entre metadados foram removidas e também foram agrupados semanticamente, gerando no final um conjunto de 21 metadados principais, sendo que a maioria desses metadados se subdividem em outros, para uma catalogação mais detalhada (Vegi, 2012).

2.5. FERRAMENTAS

2.5.1. Flexible Extensible Digital Object Repository Architecture

O *Flexible Extensible Digital Object Repository Architecture* (Fedora) (Arquitetura Flexível e Extensível de repositório de objetos digitais) é um produto da *Fedora-Commons*, uma organização sem fins lucrativos que tem a missão de prover tecnologias que permitam armazenamento seguro e acesso a conteúdos digitais. O Fedora, como todos os produtos dessa organização, é *open-source* (Davis, 2011a).

O Fedora serve como um repositório para conteúdos digitais e, apesar de ser possível usá-lo como um servidor independente de conteúdo, ele foi projetado para trabalhar em conjunto com outras aplicações (Davis, 2011a).

Na FIGURA 6 tem-se a imagem inicial do Fedora, quando acessado via o aplicativo Tomcat.


		Fedora Repository Information View	
Repository Name: Fedora Repository			
Base URL:	http://localhost:8080/fedora		
Version:	3.6.2		
PID Namespace:	changeme		
PID Delimiter:	:		
Sample PID:	changeme:100		
Retain PID Namespace:	*		
OAI Namespace:	example.org		
OAI Delimiter:	:		
Sample OAI Identifier:	oai:example.org:changeme:100		
Sample Search URL:	http://localhost:8080/fedora/objects		
Sample Access URL:	http://localhost:8080/fedora/objects/demo:5		
Sample OAI URL:	http://localhost:8080/fedora/oai?verb=Identify		
Admin Email:	bob@example.org		
Admin Email:	sally@example.org		

FIGURA 6 – Tela inicial do Fedora

Fonte: Próprio Autor

O Fedora trabalha com o conceito de objetos digitais, e todos os serviços providos pelo Fedora são acessados através de *Web Services*, que são definidos em WSDL, através do protocolo SOAP (Davis, 2011a).

2.5.1.1. **Fedora Digital Object**

No Fedora, um conjunto de itens de conteúdo são denominados como *Fedora Digital Object* (FDO), ou objetos digitais, que são formados por um conjunto de *Datastreams*. Cada objetos digital possui, basicamente, as seguintes características (Davis, 2012a):

- *Persist Identifier* (PID): Um identificador para o FDO. Esse identificador deve ser único em todo o repositório local;
- *Dublin Core* (DC): Uma descrição básica do FDO;
- *Datastreams*: Um elemento que armazena o conteúdo. Cada *datastream* pode conter qualquer tipo de conteúdo.

2.5.1.2. Content Model

No Fedora, um Content Model (CModel) é definido como um modelo padrão que descreve as características básicas de algum objeto digital, incluindo todas as possíveis características que são necessárias para permitir o armazenamento e recuperação do conteúdo, informações semânticas, comportamentais e estruturais, como também um *CModel* pode incluir informações sobre permissões, exclusões e relacionamentos com outros objetos digitais requeridos (Davis, 2011b).

No Fedora, todos os objetos digitais estão interligados com algum CModel, seja um CModel criado pelo usuário ou o CModel genérico do Fedora. Através do CModel, o Fedora sabe quais são os serviços (operações) que estão disponíveis para os objetos daquele CModel (Davis, 2011a).

2.5.2. Drupal

Um CMS que permite a criação e manutenção de diferentes tipos de websites sem a necessidade de conhecimento de qualquer linguagem de programação (Peck 2013) e por ser *open-source*, ele é distribuído sob a licença GPL (Drupal), portanto ele é livre para acesso, alterações e distribuições (Weber, 2004).

A maioria dos CMS assumem o que será feito sobre eles, e essas suposições, ou pressupostos, são difíceis de serem sobrescritas. Já os *Frameworks* em contrapartida são como provedores de materiais brutos, exigindo conhecimento de linguagens de programação para serem usados (Hunter, 2013).

No Drupal, o desenvolvimento feito sobre ele é como a junção de diversos blocos que no final serão o *Web site* (Hunter, 2013) (Byron, Berry, & Bondt, 2012).

O CMS Drupal possui características como: confiabilidade, personalização e extensibilidade (Byron et al., 2012), robustez, eficiência, flexibilidade, tornando-o apto para diversos fins: Portais, *Websites* pessoais, *Websites* corporativos, Diretórios de recursos e sites internacionais (Mercer, 2006)

Todas essas características, fazem com que o CMS Drupal seja perfeito para a solução proposta neste trabalho, pois ele proporciona um ambiente robusto que pode ser alterado com facilidade e, por ser *Open-Source*, pode ter seu código-fonte adaptado para atender a alguma eventual necessidade, se isso se fizer necessário.

Na FIGURA 7, tem-se a imagem do CMS Drupal, logo após sua instalação.



FIGURA 7 – Tela inicial do Drupal

Fonte: Próprio Autor

2.5.2.1. Módulos

O Drupal trabalha com o conceito de módulos, que possuem o objetivo de estender e customizar suas funcionalidades e, caso algum módulo não exista, este pode ser criado por qualquer pessoa e compartilhado na comunidade de usuários do Drupal (Drupal).

Os módulos são arquivos que contêm, em forma código PHP, um conjunto de funcionalidades que o Drupal sabe como usar. Toda a administração e funcionalidades vem de módulos (Byron et al., 2012).

McCourt (2011), lista quatro tipos de módulos do Drupal:

- Núcleo: Obrigatórios para o Drupal funcionar. Não podem ser desativados;
- Núcleo opcional: Já vem instalados e ativados por padrão, mas podem ser desativados;
- Contribuídos: Módulos providos pela comunidade do Drupal; e,
- Customizados: Módulos desenvolvidos por você que, talvez, não faça parte da comunidade do Drupal.

2.5.3. Islandora *Framework*

Islandora é um framework *open-source* desenvolvido pela Universidade de Prince Edward Island Robertson Library, em 2006 (Stapelfeldt, 2013).

O projeto Islandora combina e aproveita o poder do CMS Drupal e o software *Repository* Fedora para criar um robusto sistema de gerenciamento de ativos digitais que podem ser utilizados para atender aos requisitos de colaboração a curto e longo prazo da administração de dados digital (Stapelfeldt, 2013).

O núcleo de tecnologias do Islandora é composto pelo *Repository* Fedora, o CMS Drupal e o indexador Solr (Stapelfeldt, 2013).

2.5.3.1. *Solution Pack*

Solution Pack (SP) é um pequeno módulo Drupal que oferece um conjunto pré-definido, composto por um Modelo de Conteúdo (CModel), formulários e visualizações baseados na comunidade de usuários Islandora, e apesar de cada SP possuir um conjunto pré-definido de conteúdo, ele pode ser modificado para atender as necessidades em particular de cada coleção de objetos ou instituição (Stapelfeldt, 2013).

2.5.3.2. *Drupal servlet filter*

Segundo Wilcox (2013a), o *Drupal servlet filter* “permite que o repositório Fedora para usar banco de dados do Drupal para autenticação, incluindo a integração com funções de usuário do Drupal”.

2.5.4. *Fedora Generic Search*

O Fedora *Generic Search* (GSearch) faz parte do *framework* de serviços do Fedora, criado por Gert Schmeltz Pedersen na *Technical University of Denmark* (Universidade Técnica da Dinamarca) com a ajuda de diversos membros da comunidade Fedora (Davis, 2012b).

Davis (2012b), cita como características do GSearch:

- Indexação de arquivos do Fedora;
- Busca por termos indexados;
- Suporte aos plug-ins das máquinas de busca: Solr, Lucene e Zebra;

- Capacidade de rodar em um servidor diferente à parte;
- Capacidade de integrar diversos ambientes Fedora; e,
- Capacidade de atualizar vários indexes em paralelo.

O GSearch é utilizado pelo *Islandora Framework* para manter a indexação de documentos atualizada (Wilcox, 2013b).

A FIGURA 8 mostra a tela inicial do GSearch que, após de instalado e configurado, pode ser acessado localmente através do endereço <http://localhost:8080/fedoragsearch/rest>.



FIGURA 8 – Tela inicial do GSearch

Fonte: Próprio Autor

2.5.5. Apache Solr

O Solr é um servidor de busca independente escrito em Java que usa em seu núcleo a biblioteca de busca Java Lucene para indexar documentos. Os documentos são inseridos (indexados) via XML, JSON, CSV ou HTTP. A consulta é feita via HTTP GET e o conteúdo é recuperado como XML, JSON, CSV ou binário (Solr, 2013).

O Solr, FIGURA 9, representa uma importante aplicação, de terceiros, utilizada no Islandora, pois permite que o Islandora procure por todo o conteúdo, armazenado no repositório Fedora, pelos indexes feito pelo Solr, gerando resultado

rápidos e relevantes (Stapelfeldt, 2013), além disso o Islandora utiliza o Solr para permitir indexação e busca flexível e configurável (Wilcox, 2013b).



FIGURA 9 – Tela Inicial do Solr
Fonte: Próprio Autor

A FIGURA 9 mostra a tela administrativa do Solr, que pode ser acessado via tomcat, ou diretamente via `http://localhost:8080/solr/admin/`, caso esteja acessando localmente.

As informações apresentadas até aqui, fornecem condições suficientes para se entender todo o processo de desenvolvimento da solução proposta neste trabalho, a seguir tem-se a metodologia que descreve todo este processo de desenvolvimento.

3. METODOLOGIA

Para se atingir o objetivo proposto neste trabalho, isto é, viabilizar tecnologicamente a APRI, foram necessários alguns passos, que consistem na escolha do ambiente de instalação das ferramentas necessárias para a realização deste trabalho, na obtenção, integração e customização dessas ferramentas, cada um desses passos será descrito a seguir.

3.1. O CONTEXTO ONDE O TRABALHO FOI REALIZADO

O primeiro passo foi a escolha do ambiente para a realização do trabalho, A escolha do ambiente para a realização do trabalho foi feita de forma cuidadosa, pois um ambiente de instalação problemático poderia comprometer a conclusão deste trabalho, sendo assim, a solução proposta neste trabalho se deu em um computador pessoal com as principais características de hardware:

- Processador Intel® Core™ i-3-2100 (3.10GHz); e,
- Memória RAM 4GB.

A solução apresentada, ainda se deu em um ambiente de desenvolvimento UNIX, com as seguintes características:

- Sistema Operacional Debian Squeeze;
- Servidor *web* Apache 2.2;
- PHP 5.3; e,
- Gerenciador de Banco de Dados MySQL 5.2.

Apesar da solução proposta neste trabalho, ter sido desenvolvida no ambiente apresentado acima, ela também foi testada com sucesso em ambiente de

desenvolvimento *Windows 7*, e em outros ambientes de desenvolvimentos UNIX, contendo sistemas operacionais Ubuntu 13.10 (64/32bits) e Ubuntu 13.04.

3.2. OBTENÇÃO DAS FERRAMENTAS

Todas as ferramentas são gratuitas e podem ser adquiridas através dos *sites* oficiais de cada uma delas. Abaixo, tem-se uma lista com as ferramentas e seus respectivos endereços na *web*.

- Fedora *Repository* (versão 3.6.2): <http://downloads.sourceforge.net/fedora-commons/fcrepo-installer-3.6.2.jar>
- Solr (versão 3.6.1): <http://archive.apache.org/dist/lucene/solr/3.6.1/apache-solr-3.6.1.zip>
- Gsearch (versão 2.6): <http://downloads.sourceforge.net/fedora-commons/fedoragsearch-2.6.zip>
- Drupal (versão 6): <http://ftp.drupal.org/files/projects/drupal-6.28.zip>
- Islandora *Framework* e Solutions Pack (versão 6.x-13.1.x): <https://wiki.duraspace.org/display/ISLANDORA6131/Release+Notes+and+Downloads>
- Drupal Servlet Filter (versão 3.6.2): <https://wiki.duraspace.org/download/attachments/34638844/fcrepo-drupalauthfilter-3.6.2.jar?version=1&modificationDate=1364905404848&api=v2>

3.3. INSTALAÇÃO

A instalação das ferramentas foram realizadas conforme sugerido pelas documentações oficiais do Fedora e do Islandora, essas documentações estão disponíveis nos sites oficiais dos mesmo.

3.4. CUSTOMIZAÇÕES DAS FERRAMENTAS

Após a instalação de cada uma das ferramentas, diversas customizações foram feitas para que a inserção, a recuperação e exibição de dados fosse eficiente. Todo o processo de customização passou pelos passos descritos nos tópicos a seguir.

3.4.1. Drupal

Foi necessário alterar dois arquivos do Drupal, essas alterações permitem o Drupal tratar corretamente a exibição do objetos DC2AP, mantendo a exibição padrão de outros objetos.

3.4.1.1. Arquivo ObjectDetails.inc

O arquivo ObjectDetails.inc é responsável por definir a exibição padrão dos objetos no Islandora.

Neste arquivo, a função `fedora_repository_object_details_table` foi alterada para que, quando receber um arquivo que contenha o *datastream* DC2AP, duas ações fossem feitas.

A primeira ação consiste na coleta dos metadados do objeto, contidos no *datastream* DC2AP, e esses metadados fossem organizados em uma estrutura de

array. Para que essa organização fosse feita com sucesso, foram criadas duas funções:

- *recupera_metadados* : que percorre, recursivamente, o *array* que contém os metadados, recupera cada metadados nesse *array*, pelo *namespace* e envia para função *organiza_dados_em_array*;
- *organiza_dados_em_array*: que cria um *array* com o metadados recebido, esse *array* é composta apenas por um metadados e o nome de uma classe em *Cascating Style Sheets* (CSS).

A segunda ação consiste em receber este *array* que foi criado com os metadados reorganizados, e encaminhar para a função que tratará a exibição desse objeto, a alteração feita nessa função consistiu em adicionar uma estrutura de condição, para que o funcionamento padrão do Drupal não fosse alterado, quando qualquer outro tipo de objeto fosse inserido.

3.4.1.2. Arquivo *theme.inc*

Esse arquivo controla praticamente toda a interface de exibição do Drupal e foram feitas diversas alterações, iniciando pela função *theme*, para que ao receber um *array* contendo o índice DC2AP, o *array* seja encaminhado para a função que tratará da forma correta seu conteúdo.

De forma análoga à função *fedora_repository_object_details_table*, do arquivo *ObjectDetails.inc*, essa primeira alteração foi feita criando uma estrutura de condição, para que o comportamento padrão do Drupal, com relação aos outros objetos que não fazem parte da APRI, não fosse alterado.

Foi criada a função *theme_table_dc2ap*, que é a responsável por todo o tratamento de dados do *array* que contém os metadados DC2AP, esta função é uma cópia, da função *theme_table* (presente neste mesmo arquivo), que foi adaptada.

Esta função retorna uma tabela com todos metadados, com seus respectivos conteúdos, para ser exibida para o usuário.

Para essa função funcionar corretamente foram criadas as funções:

- Função `procura_conteudo_dc2ap`: Que recebe um array de metadados e procura, recursivamente, pelo conteúdo desses metadados, ignorando os que são irrelevantes para o usuário. O retorno desse função consiste em um array, sem os metadados que não irrelevantes para o usuário;
- Função `_theme_table_cell_dc2ap` que é uma cópia da função `_theme_table_cell` (nativa do Drupal) e tem como objetivo criar cada linha da tabela, inserindo as classes e estilos CSS necessário para a exibição final para o usuário esteja correta. Esse função traduz os metadados para língua portuguesa todos os metadados DC2AP, por meio da função `traduz_metadado_para_exibicao_dc2ap`.

3.4.2. GSearch

No GSearch foi alterado o arquivo `foxmlToSolr.xml`, que tem a função de receber os metadados, do objeto inserido no Fedora e enviá-los para o Solr, para serem indexados.

3.4.3. Solr

No Solr foram alterados os arquivos `schema.xml` e `solrconfig.xml`, conforme descrito a seguir:

- `schema.xml`: uma das responsabilidades desse arquivo é a indexação dos documentos no Solr. Esse arquivo foi alterado para armazenar quaisquer campos que ele receber, assim, todos os metadados do DC2AP serão indexados;

- `solrconfig.xml`: uma das responsabilidades deste arquivo são as buscas no repositório de indexes do Solr, que são controladas pelos RequestHandlers. Neste arquivo foi adicionado um RequestHandler para as buscas pelos metadados DC2AP.

3.5. Criação do Pandora SP

Para finalizar o trabalho, foi criado um Solution Pack para que os dados fossem armazenados de forma correta no Fedora. Este SP é uma cópia de um SP existente, chamado de PDF SP, que foi alterado para atender as nossas necessidades. Foi criado o seguinte conteúdo para o Pandora SP:

- `DC2AP_Form`: Um formulário para a inserção de padrões de análise, através do Drupal;
- `PANDORA_COLLECTION_POLICY`: Arquivo que registra a qual CModel, pertence o objeto digital;
- `STRICT_SP_PANDORACM`: Um modelo de conteúdo para os objetos digitais que conterão os padrões de análises;

3.6. Testes

Para testar as funcionalidades da solução que está sendo proposta neste trabalho, foram realizadas as seguintes operações, na ordem que se segue:

- Instalação do Pandora SP;
- Inserção dos 5 padrões de análises, que estão disponíveis para download em http://www.dpi.ufv.br/projetos/apri/?page_id=183;
- Exibição dos padrões de análises inseridos;
- Busca pelo padrão de análise *Organization Hierarchies*;

- Exibição dos resultados da busca.

Feito todos estes testes, foi possível avaliar os resultados finais e definitivos da viabilização tecnológica da APRI, proposta deste trabalho, os resultados obtidos, como a análise deles, serão abordados com mais detalhes na seção seguinte.

4. ANÁLISE DOS RESULTADOS

Após a realização dos testes, foi possível obter resultados positivos no que se refere ao portal proposto neste trabalho. Os resultados obtidos são apresentados, e analisados, nas seções a seguir, também serão apresentadas algumas imagens onde poderá ser observado que o portal possui uma interface limpa e amigável, apresentada pelo CMS Drupal.

4.1. Inserção dos padrões de análises

O portal disponibiliza para o usuário um formulário para a catalogação dos padrões de análise, FIGURA 10, como o formulário foi feito com base no DC2AP, temos então um conjunto campos que permitem uma catalogação detalhada de qualquer padrão de análise, por exemplo, os padrões podem ser catalogados com informações sobre as consequências positivas e negativas do uso dos padrões, as alterações que ocorreram no padrão desde a primeira versão e também diagramas de classes que fornecem uma descrição mais técnica do padrão.

Coleção PANDORA

[Ver](#) [Add](#) [Manage This Collection](#) [Object Details](#)

Título:

Título(s) Alternativo(s):

Criador(es):

Assunto(s):

 Empresas Hierarquia Estrutura Organizacional

Descrição

Problema(s):
 Existem muitos sistemas onde precisamos gerir a hierarquia de uma organização, cadastrando suas filiais e vinculando-as de acordo com as regras da hierarquia. Como podemos representar este processo de uma forma geral e abstrata?

FIGURA 10 – Padrão de análise sendo documentado
 Fonte: Próprio Autor

Na FIGURA 10 vê-se o formulário de inserção de padrões de análise, sendo exibido corretamente e preenchido com informações do padrão de análise *Organization Hierarchies*.

4.2. Exibição dos padrões de análises inseridos

A visualização dos padrões de análise, FIGURA 13, é apresentada em uma interface leve e agradável, onde o usuário pode visualizar as informações de forma organizada. A organização dos metadados na exibição também é algo a ser notado, visto que metadados semelhantes são agrupados em categorias para melhor visualização e compreensão do que está sendo descrito.

Organization Hierarchies	
Ver Editar	
Metadata (DC2AP)	
Título	Organization Hierarchies
Criador	Martin Fowler
	Empresas
	Hierarquia
Assunto	Estrutura Organizacional
	Filiais
Problema	Existem muitos sistemas onde precisamos gerir a hierarquia de uma organização, cadastrando suas filiais e vinculando-as de acordo com as regras da hierarquia. Como podemos representar este processo de uma forma geral e abstrata?
Motivos	Uma hierarquia organizacional possui subdivisões como Unidades Operacionais, Regiões, Divisões e Escritórios de Venda. Unidades Operacionais são divididas em Regiões. Regiões são divididas em Divisões. Divisões são divididas em Escritórios de Venda.
Descrição	Precisamos prover uma solução fácil de ser alterada, pois organizações sofrem mudanças na sua hierarquia ao decorrer do tempo.
Exemplo	Um sistema de gestão de uma empresa multinacional como, por exemplo, a Microsoft. Um sistema de gestão de uma empresa nacional que tenha várias ramificações espalhadas por todo o território do país, por exemplo, a rede de lanchonete Giraffas.
	..

FIGURA 11 – Exibição do padrão de análise *Organization Hierarchies*
Fonte: Próprio Autor

Na FIGURA 11 vê-se o padrão de análise *Organization Hierarchies* já inserido, e sendo exibido, pode-se observar que a interface é simples e as informações são facilmente interpretadas.

4.3. Busca pelo padrão de análise *Organization Hierarchies*

A ferramenta de pesquisa é de interface simples, onde o usuário tem duas opções de busca:

- Busca Simples: O usuário digita a informação e a pesquisa é feita nos metadados que a administrador do portal especificar; e,

- Busca Detalhada: O usuário digita a informação e a pesquisa é feita nos metadados que o usuário especificar, essa lista de metadados disponíveis para a busca e feita pelo administrador do portal.

O resultado da busca é satisfatório e retornado de forma rápida, já que se beneficia da indexação de dados providos pelo Apache Solr.

The screenshot displays a web interface for 'Organization Hierarchies'. On the left, there is a metadata table and a description section. On the right, there is a search panel with two sections: 'Busca Simples' and 'Busca Detalhada'. The 'Busca Simples' section has a text input field containing 'Organization Hierarchies' and a 'buscar' button. The 'Busca Detalhada' section has a dropdown menu set to 'Título', an empty text input field, and a 'buscar' button. A red rectangular box highlights the search input fields and buttons in the 'Busca Simples' and 'Busca Detalhada' sections.

Organization Hierarchies	
Ver Editar	
Metadada (DC2AP)	
Título	Organization Hierarchies
Criador	Martin Fowler
	Empresas
	Hierarquia
Assunto	Estrutura Organizacional
	Filiais
Problema	Existem muitos sistemas onde precisamos gerir a hierarquia de uma organização, cadastrando suas filiais e vinculando-as de acordo com as regras da hierarquia. Como podemos representar este processo de uma forma geral e abstrata?
Motivos	Uma hierarquia organizacional possui subdivisões como Unidades Operacionais, Regiões, Divisões e Escritórios de Venda. Unidades Operacionais são divididas em Regiões. Regiões são divididas em Divisões. Divisões são divididas em Escritórios de Venda.
Descrição	Precisamos prover uma solução fácil de ser alterada, pois organizações sofrem mudanças na sua hierarquia ao decorrer do tempo. Um sistema de gestão de uma empresa multinacional como, por exemplo, a Microsoft.
Exemplo	Um sistema de gestão de uma empresa nacional que tenha várias ramificações espalhadas por todo o território do país, por exemplo, a rede de lanchonete Giraffas.

FIGURA 12 – Local do serviço de busca por padrões de análise
Fonte: Próprio Autor

Na FIGURA 12 vê-se, a direita, em destaque (quadro vermelho), o campo para se realizar a busca por padrões de análise.

5. Conclusão

Mediante as análises dos resultados expostos, pode-se observar que o portal desenvolvido fornece ao usuário a possibilidade de catalogar padrões de análise de forma simples e detalhada, o catálogo de padrões de análise pode ser acessado via *Web services* e o usuário também tem um mecanismo de busca para agilizar o processo de procura por algum padrão de análise.

Diante de tais possibilidades chegamos à conclusão que as expectativas concernentes ao portal da APRI foram satisfeitas, e que o portal se torna mais uma alternativa no que se refere a catalogação e divulgação de padrões de análise, eliminando assim, diversos problemas, referentes aos padrões de análise, abordados no decorrer deste trabalho.

6. Trabalho futuro

Como trabalho futuro, fica a sugestão de desenvolver um *software* que através dos *web services* providos pelo Fedora Repository pudesse acessar os padrões de análise e recuperar informações de forma semântica;

7. REFERÊNCIAS

- Alves, M. das D. R., & Souza, M. I. F. (2007). **Estudo de correspondência de elementos metadados: Dublin core e MARC21**. *Revista Digital de Biblioteconomia e Ciência da Informação*, 4, 20–38.
- Blaimer, N., Bortfeldt, A., & Pankratz, G. (2010). **Patterns in Object-Oriented Analysis** (No. 451). *Analysis* (p. 80). Hagen.
- Braude, Eric J. **Projeto De Software: Da Programação À Arquitetura: Uma Abordagem Baseada Em Java**. Porto Alegre, RS: Bookman, 2005
- Byron, A., Berry, A., & Bondt, B. De. (2012). **Using Drupal** (p. 496).
- Cerami, E. (2002). **Web Services Essentials** (1st ed., p. 304). O'Reilly.
- Coad, P. (1992). **Object-oriented patterns**. *Communications of the ACM*, 35(9), 152–159. doi:10.1145/130994.131006
- Davis, D. (2011a). **Tutorial 1 - Introduction to Fedora**. Retrieved October 19, 2013, from <https://wiki.duraspace.org/display/FEDORACREATE/Tutorial+1+-+Introduction+to+Fedora>
- Davis, D. (2011b). **Content Model Architecture**. Retrieved October 20, 2013, from <https://wiki.duraspace.org/display/FEDORA36/Content+Model+Architecture>
- Davis, D. (2012a). **Fedora Digital Object Model**. Retrieved October 20, 2013, from <https://wiki.duraspace.org/display/FEDORA36/Fedora+Digital+Object+Model>
- Davis, D. (2012b). **Generic Search Service 2.2**. Retrieved October 24, 2013, from <https://wiki.duraspace.org/display/FCSVCS/Generic+Search+Service+2.2>
- DCMI. **History of the Dublin Core Metadata Initiative**. Retrieved October 17, 2013, from <http://dublincore.org/about/history/>

Drupal. **History**. Retrieved October 20, 2013, from <https://drupal.org/about/history>

Drupal. **About Drupal**. Retrieved October 20, 2013, from <https://drupal.org/about>

Drupal. **Modules**. Retrieved October 20, 2013, from <https://drupal.org/project/modules>

Fawcett, J., Ayers, D., & Liam R.E., Q. (2012). **Beginning XML** (5th ed., p. 828).

Fowler, M. (1996). **Analysis patterns: reusable object models. Concurrent Engineering** (1st ed., Vol. 3, p. 361). Indianapolis: Pearson Education.

Fraser, S. R. G. (2002). *Real-World ASP . NET — Building a Content Management System* (Vol. 2002, p. 522).

Gamma, Erich, et al. **Padrões de Projeto: Soluções Reutilizáveis de Software Orientado a Objetos**. Porto Alegre: Bookman, 2002. 364 p.

Hamza, H. S. (2002). **A foundation for building stable analysis patterns**. Citeseer.

Hauschildt, S. (2010). **CMS Made Simple 1.6: Beginner's Guide** (p. 366).

Hunter, L. (2013). **The Drupal overview**. Retrieved October 20, 2013, from <https://drupal.org/getting-started/before/overview>

Krueger, C. W. (1992). **Software reuse**. *ACM Computing Surveys*, 24(2), 131–183. doi:10.1145/130844.130856

Lucrédio, D. (2009). **Uma Abordagem Orientada a Modelos para Reutilização de Software**. Universidade de São Paulo.

McCourt, C. (2011). **Drupal: The Guide to Planning and Building Websites**. Wiley Publishing, Inc.

Mercer, D. (2006). **Drupal - Creating Blogs, Forums, Portals, and Community Websites** (p. 267). Birmingham, UK : Packt Publishing, c2006.

Newcomer, E. (2004). ***Understanding Web Services: XML, WSDL, SOAP, and UDDI*** (3rd ed., p. 332).

Pantoquilha, M., Raminhos, R., & Araújo, J. (2003). ***Analysis Patterns Specifications: Filling the Gaps***. *VIKING PLOP* (Vol. 18, pp. 1–13). Bergen, Norway.

PAULA, Wilson de Pádua F. **Manual do Engenheiro de Software**. 2000.

Peck, S. (2013). ***Understanding Drupal***. Retrieved June 11, 2013, from <https://drupal.org/documentation/understand>

Peters, James F. PEDRYCZ, Withold. **Engenharia de Software: Teoria e Prática**. Rio de Janeiro: Campus, 2001. 602 p.

R. E. Quin, Liam. ***XML Essentials***. Retrieved January 15, 2013, from <http://www.w3.org/standards/xml/core>

Redding, J. (2010). ***Beginning Drupal***. Wiley Publishing, Inc.

REZENDE, Denis A. **Engenharia de Software e Sistemas de Informação**. 3ª edição. Rio de Janeiro: BRASPORT Livros e Multimídia LTDA, 2005.

Solr. (2013). **Apache Solr**. Retrieved October 20, 2013, from <http://lucene.apache.org/solr/index.html>

Sommerville, I. (2011). ***Software engineering*** (pp. 1–773).

Stapelfeldt, K. (2013). ***Introducing Islandora***. Retrieved June 11, 2013, from <https://wiki.duraspace.org/display/ISLANDORA711/Chapter+1+-+Introducing+Islandora>

Vegi, L. F. da M. (2012). **Um perfil de metadados Dublin Core para documentar padrões de análise em uma infraestrutura de reuso**. Universidade Federal de Viçosa.

Vegi, L. F. da M., Peixoto, D. A., Lisboa Filho, J., & Oliveira, A. D. P. (2012). **An Infrastructure Oriented for Cataloging Services and Reuse of Analysis Patterns**. *Workshop on Reuse in Business Process Management*, 1–6.
doi:10.1007/978-3-642-28115-0_32

Weber, S. (2004). **The success of open source** (p. 320). Harvard University Press.

Wilcox, D. (2013). Chapter 9 - **Enabling Indexing & Searching with Solr**. Retrieved October 20, 2013, from
<https://wiki.duraspace.org/pages/viewpage.action?pageId=34644675>

**Apêndice 1 – Alteração feita na função
fedora_repository_object_details_table do arquivo ObjectDetails
alterada.**

```

$rows = array();
if (array_key_exists('DC2AP', $item->datastreams)) {
    /** PANDORA */
    foreach ($simplexml->getNamespaces(TRUE) as $ns) {
        foreach ($simplexml->children($ns) as $child) {
            $rows[$child->getName()] = organiza_dados_em_array($child);
            if ((string) $child == "") {
                $rows[$child->getName()]['valor']['data'] = recupera_metadados($child);
            }
        }
    }
}
return theme('table', $headers, $rows, array(), 'DC2AP');
} else {
    /** FUNCAO NATIVA */
    foreach ($simplexml->getNamespaces(TRUE) as $ns) {
        foreach ($simplexml->children($ns) as $child) {
            $rows[] = array(
                array(
                    'data' => $child->getName(),
                    'class' => 'dc-tag-name',
                ),
                array(
                    'data' => (string) $child,
                    'class' => 'dc-content',
                ),
            );
        }
    }
    return theme('table', $headers, $rows, array('class' => 'dc-table'));
}

```

Apêndice 2 – Funções adicionadas no arquivo ObjectDetails.inc

```
function recupera_metadados($simplexml, $nivel = FALSE) {
    foreach ($simplexml->getNamespaces(TRUE) as $ns) {
        foreach ($simplexml->children($ns) as $child) {
            $dados = organiza_dados_em_array($child);
            if (strlen($dados['valor']['data']) > 0) {
                $metadados_internos[$child->getName()]['data'][] = $dados['valor']['data'];
                $metadados_internos[$child->getName()]['class'] = $dados['valor']['class'];
            } else {
                $metadados_internos[$child->getName()] = recupera_metadados($child, 2);
            }
        }
    }
    return $metadados_internos;
}

function organiza_dados_em_array($child) {
    if (!$metadados_internos) {
        return array(
            'campo' =>
            array(
                'data' => $child->getName(),
                'class' => 'dc-tag-name',
            ),
            'valor' =>
            array(
                'data' => (string) $child,
                'class' => 'dc-content',
            ),
        );
    }
}
```

Apêndice 3 – Alteração feita na função theme, localizada no arquivo theme.inc

```
if (isset($info['function'])) {
    // The theme call is a function.
    if ($args[3] == 'DC2AP' && $info['function'] == 'theme_table') {
        /** pandora */
        $output = call_user_func_array('theme_table_dc2ap', $args);
    } else {
        $output = call_user_func_array($info['function'], $args);
    }
} else {
    // The theme call is a template.
    $variables = array(
        'template_files' => array()
    );
};
```

Apêndice 4 – Funções adicionadas no arquivo theme.inc

```

function theme_table_dc2ap($header, $rows, $attributes = array(), $caption = NULL) {
  // Add sticky headers, if applicable.
  if (count($header)) {
    drupal_add_js('misc/tableheader.js');
    // Add 'sticky-enabled' class to the table to identify it for JS.
    // This is needed to target tables constructed by this function.
    $attributes['class'] = empty($attributes['class']) ? 'sticky-enabled' : ($attributes['class'] . ' sticky-enabled');
  }
  $output = '<table' . drupal_attributes($attributes) . ">\n";

  // Format the table header:
  if (count($header)) {
    $sts = tablesort_init($header);
    // HTML requires that the thead tag has tr tags in it followed by tbody
    // tags. Using ternary operator to check and see if we have any rows.
    $output .= (count($rows) ? '<thead><tr>' : '<tr>');
    foreach ($header as $cell) {
      $cell = tablesort_header($cell, $header, $sts);
      $output .= _theme_table_cell($cell, TRUE);
    }
    // Using ternary operator to close the tags based on whether or not there are rows
    $output .= (count($rows) ? "</tr></thead>\n" : "</tr>\n");
  } else {
    $sts = array();
  }
  // Format the table rows:
  if (count($rows)) {
    $output .= "<tbody>\n";
    $flip = array('even' => 'odd', 'odd' => 'even');
    $class = 'even';
    foreach ($rows as $number => $row) {
      $attributes = array();
      // Check if we're dealing with a simple or complex row
      if (isset($row['data'])) {
        foreach ($row as $key => $value) {
          if ($key == 'data') {
            $cells = $value;
          } else {
            $attributes[$key] = $value;
          }
        }
      } else {
        $cells = $row;
      }
      if (count($cells)) {
        // Add odd/even class
        $class = $flip[$class];
        if (isset($attributes['class'])) {
          $attributes['class'] .= ' ' . $class;
        } else {
          $attributes['class'] = $class;
        }
      }
      $conjunto = array();
    }
  }
}

```



```

foreach ($cells as $cell) {
    if (is_array($cell['data'])) { //verifica se é um array
        $cell = procura_conteudo_dc2ap($cell['data'], 0, 0);
    } elseif (gettype($cell['data']) == 'NULL') {
        $cell['data'] = "";
    }

    $conjunto[] = $cell;
}
foreach ($conjunto as $nome => $parte) {
    $conjunto[1] = calcula_rowspan($conjunto[1]);
    $continue = 0;
    if (is_array($conjunto[1]) && array_key_exists('data', $conjunto[1]) &&
gettype($conjunto[1]['data']) != 'string') {
        $rowspan = array_pop($conjunto[1]) + 1;
        $continue = 1;
    } elseif (is_array($conjunto[1]) && !array_key_exists('data', $conjunto[1])) {
        $rowspan = sizeof($conjunto[1]) + 1;
        $continue = 1;
    }
    if ($continue) {
        foreach ($conjunto[1] as $chave => $conjunto_parte) {
            if (is_array($conjunto_parte) && array_key_exists('rowspan', $conjunto_parte))
{
                $rowspan += $conjunto_parte['rowspan'];
            }
        }
        $conjunto[0]['rowspan'] = $rowspan;
    }
}
// Build row
$output .= '<tr>';

foreach ($conjunto as $array) {
    $output .= _theme_table_cell_dc2ap($array, $attributes);
}
$output .= "</tr>\n";
}
}
$output .= "</tbody>\n";
}
$output .= "</table>\n";
return $output;
}

```

```

function calcula_rowspan($array) {
    if (is_array($array) && gettype($array['data']) != 'string') {
        foreach ($array as $metadado => $parte) {
            if ($metadado == 'data') {
                $tam = sizeof($array[$metadado]);
                $array['rowspan'] = $tam;
            } elseif (is_array($parte) && array_key_exists('data', $parte)) {
                $tam = sizeof($parte['data']);
                $array[$metadado]['rowspan'] = $tam;
            } elseif (gettype($parte) == 'array') {
                $tam = sizeof($parte);
                $array_recursivo = calcula_rowspan($parte);
                foreach ($array_recursivo as $parte_recursivo) {
                    if (is_array($parte_recursivo) && array_key_exists('rowspan', $parte_recursivo)) {
                        $tam += $parte_recursivo['rowspan'];
                    }
                }
                $array[$metadado] = $array_recursivo;
                $array[$metadado]['rowspan'] = $tam - 1;
            }
        }
    }
    return $array;
}

function procura_conteudo_dc2ap($array, $rowspan) {
    $metadados_ignorados = array('Bag', 'Seq', 'li', 'W3CDTF', 'ISO639-3', 'Alt', 'value', 'data', 'class');
    if (is_array($array)) {
        $chaves = array_keys($array);
        if (in_array('data', $chaves)) {
            $rowspan = sizeof($array['data']);
            return $array;
        } else {
            $flag_metadado_importante = 1;
            foreach ($chaves as $i_chaves => $chave) {
                if (in_array($chave, $metadados_ignorados)) {
                    $flag_metadado_importante = 0;
                    break;
                }
            }
            foreach ($chaves as $num_chave => $nome_chave) {
                if ($flag_metadado_importante) {
                    if (is_array($array[$chaves[$num_chave]])) {
                        $array_atual = $array[$chaves[$num_chave]];
                    }
                    $aux = procura_conteudo_dc2ap($array[$chaves[$num_chave]], $rowspan);
                    $metadados[$nome_chave] = $aux;
                } else {
                    $metadados = procura_conteudo_dc2ap($array[$chaves[$num_chave]], $rowspan);
                }
            }
        }
    } else {
        return $array;
    }
    return $metadados;
}

```

```

function _theme_table_cell_dc2ap($cell, $attributesExt = FALSE, $header = FALSE) {
  $attributes = "";

  if (isset($cell['class']) && $cell['class'] == 'dc-tag-name') {
    $cell['data'] = traduz_metadado_para_exibicao_dc2ap($cell['data']);
  }
  if (is_array($cell)) {
    if (isset($cell['data'])) {
      $data = isset($cell['data']) ? $cell['data'] : "";
      $ativa_recusividade = 0;
    } else {
      $data = $cell;
      $ativa_recusividade = 1;
    }
    $header |= isset($cell['header']);
    unset($cell['header']);
    if (isset($cell['data']) || $cell['class'] == 'dc-content') {
      unset($cell['data']);
      $cell['style'] = 'border-top:1px solid #d3e7f4; border-bottom:1px solid #d3e7f4;';
      if ($cell['class'] == 'dc-tag-name') {
        $cell['style'] .= 'font-weight:bold';
      }
      $attributes .= drupal_attributes($cell);
    }
  } else {
    $data = $cell;
  }

  if ($header) {
    $output = "<th$attributes>$data</th>";
  } else {
    if (is_array($data)) {
      if ($ativa_recusividade) {
        foreach ($data as $nome_metadado => $array) {
          $estilo = 'border-top:1px solid #d3e7f4; border-bottom:1px solid #d3e7f4;';
          if (is_array($array)) {
            $rowspan = array_pop($array) + 1;
          }
          $attributes = array('rowspan' => $rowspan,
            'style' => $estilo
          );
          unset($rowspan);
          $attributes = drupal_attributes($attributes);
          $nome_metadado = traduz_metadado_para_exibicao_dc2ap($nome_metadado);
          $output .= "<tr>";
          $output .= "<td$attributes><i>$nome_metadado</i></td>";
          $output = _theme_table_cell_dc2ap($array);
          $output .= "</tr>";
        }
      } else {
        foreach ($data as $conteudo) {
          $output .= "<tr>";
          $output .= "<td$attributes>$conteudo</td df>";
          $output .= "</tr>";
        }
      }
    } else {
      if (gettype($data) != 'NULL') {
        $output .= "<td$attributes>$data</td>";
      }
    }
  }
  return $output;
}

```

```

function traduz_metadado_para_exibicao_dc2ap($metadado) {
    $array_para_traducao = array(
        'activityStateDiagrams' => 'Diagramas de estado de atividade',
        'alternative' => 'Títulos Alternativos',
        'analysisPattern' => 'Padrão de Análise',
        'author' => 'Autor',
        'behaviour' => 'Comportamento',
        'changes' => 'Alterações',
        'classDescriptions' => 'Descrição de Classes',
        'classDiagram' => 'Diagramas de Classes',
        'collaborationSequenceDiagrams' => 'Diagrama Sequencia de Colaboração',
        'complementedBy' => 'Complementado por',
        'conflictAndResolution' => 'Conflitos e Resoluções',
        'consequences' => 'Consequencias',
        'context' => 'Contexto',
        'contributionGraph' => 'Grafico de Contribuições',
        'contributor' => 'Contribuidor',
        'coverage' => 'Segurança',
        'created' => 'Data de Criação',
        'creator' => 'Criador',
        'date' => 'Data',
        'dependenciesAndContributions' => 'Dependências e contribuições',
        'dependencyGraph' => 'Grafico de Dependências',
        'description' => 'Descrição',
        'designGuidelines' => 'Diretrizes de projeto',
        'example' => 'Exemplo',
        'format' => 'Formato',
        'formOfNotation' => 'Notação de Formulário',
        'functionalRequirements' => 'Requerimentos Funcionais',
        'hasPart' => 'Possui parte',
        'history' => 'História',
        'isDesignedWith' => 'É projetado com',
        'isPartOf' => 'Parte de:',
        'isReplacedBy' => 'Substituído por',
        'isVersionOf' => 'É versão de',
        'knownUses' => 'Usos conhecidos',
        'label' => 'Etiqueta',
        'language' => 'Idioma',
        'modelling' => 'Modelagem',
        'modified' => 'Data de Modificação',
        'motivation' => 'Motivos',
        'negative' => 'Negativos',
        'nonFunctionalRequirements' => 'Requerimentos não-funcionais',
        'note' => 'Nota',
        'participants' => 'Participantes',
        'positive' => 'Positivos',
        'priorities' => 'Prioridades',
        'problem' => 'Problema',
        'publisher' => 'Publicador',
        'reason' => 'Razão',
        'relation' => 'Relação',
        'relationshipDescriptions' => 'Descrição de Relacionamentos',
        'replaces' => 'Substituições',
        'requirements' => 'Requerimentos',
        'resultingContext' => 'Contexto resultante',
        'rights' => 'Direitos',
        'shouldAvoid' => 'Deve Evitar',
        'solutionVariants' => 'Soluções Variantes',
        'source' => 'Fonte',
        'structure' => 'Estrutura',
        'subject' => 'Assunto',
        'title' => 'Título',
        'type' => 'Tipo',
        'useCaseDiagram' => 'Diagramas de Casos de Uso',
        'xmi' => 'xmi'
    );
    return $array_para_traducao[$metadado];
}

```

Apêndice 5 – Modelo de conteúdo criado para o Pandora SP (arquivo STRICT_SP_PANDORACM.xml)

```
<content_model name="strict_pandora" xmlns="http://www.islandora.ca"  
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
xsi:schemaLocation="http://www.islandora.ca http://localhost/islandoracm.xsd">  
  
</content_model>
```

Apêndice 6 – Arquivo que define a qual modelo de conteúdo deverá ser usado pela coleção Pandora (arquivo PANDORA-COLLECTION POLICY.xml)

```
<?xml version="1.0" encoding="utf-8"?>
<collection_policy xmlns="http://www.islandora.ca"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name=""
xsi:schemaLocation="http://www.islandora.ca
http://syn.lib.umanitoba.ca/collection_policy.xsd">
  <content_models>
    <content_model name="Modelo de Conteúdo PANDORA" dsid="ISLANDORACM"
namespace="pandora:islandoraCM" pid="pandora:islandoraCM"/>
  </content_models>
  <search_terms>
  </search_terms>
  <relationship>isMemberOfCollection</relationship>
</collection_policy>
```

Apêndice 7 – Alteração feita no arquivo schema.xml para indexar metadados Pandora.

```
<dynamicField name="*" type="text" indexed="false" stored="false" multiValued="true"/>
```

Apêndice 8 – Alteração feita no arquivo solrconfig.xml para que o sistema de busca funcione.

```
<requestHandler name="PANDORA" class="solr.SearchHandler">
  <lst name="defaults">
    <str name="echoParams">explicit</str>
    <str name="fl"> Titulo,Assunto,Autor,Colaboradores,Data,Local,Descrição,PID,Idioma</str>
    <str name="q.alt">*:*</str>
    <str name="qf">dc.title^2.5 dc.subject^1.5 dc.description dc.creator^3.0 dc.contributor^5.0
PID^0.5 </str>
  </lst>
</requestHandler>
```


Apêndice 9 – Arquivo foxToSolr.xslt (responsável por recuperar os metadados DC2AP e formata-los para serem indexados pelo Solr.

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- $Id: foxmlToSolr.xslt $ -->
<xsl:stylesheet version="1.0"
xmlns="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:exts="xalan://dk.defxws.fedoragsearch.server.GenericOperationsImpl"
exclude-result-prefixes="exts"
xmlns:foxml="info:fedora/fedora-system:def/foxml#"
xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc/"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:rdaroles="http://rdvocab.info/roles/"
xmlns:rdaelem="http://rdvocab.info/Elements/"
xmlns:dc2aptp="http://purl.org/dc2ap/type/"
xmlns:dc2apft="http://purl.org/dc2ap/format/"
xmlns:dc2ap="http://purl.org/dc2ap/elements/"
xmlns:dcterms="http://purl.org/dc/terms/">

<xsl:output method="xml" indent="yes" encoding="UTF-8"/>
<xsl:param name="REPOSITORYNAME" select="repositoryName"/>
<xsl:param name="FEDORASOAP" select="repositoryName"/>
<xsl:param name="FEDORAUSER" select="repositoryName"/>
<xsl:param name="FEDORAPASS" select="repositoryName"/>
<xsl:param name="TRUSTSTOREPATH" select="repositoryName"/>
<xsl:param name="TRUSTSTOREPASS" select="repositoryName"/>
<xsl:variable name="PID" select="/foxml:digitalObject/@PID"/>

<xsl:template match="/">
<!-- The following allows only active FedoraObjects to be indexed. -->
<xsl:if test="foxml:digitalObject/foxml:objectProperties/foxml:property[@NAME='info:fedora/fedora-system:def/model#state' and @VALUE='Active']">
<xsl:if test="not(foxml:digitalObject/foxml:datastream[@ID='METHODMAP'] or
foxml:digitalObject/foxml:datastream[@ID='DS-COMPOSITE-MODEL'])">
<xsl:if test="starts-with($PID,")">
<xsl:apply-templates mode="activeFedoraObject"/>
</xsl:if>
</xsl:if>
</xsl:if>
<!-- The following allows inactive FedoraObjects to be deleted from the index. -->
<xsl:if test="foxml:digitalObject/foxml:objectProperties/foxml:property[@NAME='info:fedora/fedora-system:def/model#state' and @VALUE='Inactive']">
<xsl:if test="not(foxml:digitalObject/foxml:datastream[@ID='METHODMAP'] or
foxml:digitalObject/foxml:datastream[@ID='DS-COMPOSITE-MODEL'])">
<xsl:if test="starts-with($PID,")">
<xsl:apply-templates mode="inactiveFedoraObject"/>
</xsl:if>
</xsl:if>
</xsl:if>
</xsl:template>

```

```

<xsl:template match="/foxml:digitalObject" mode="activeFedoraObject">
  <add>
  <doc>
  <field name="PID">
  <xsl:value-of select="$PID"/>
  </field>
  <xsl:for-each
  select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:title">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc',',','title')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
  </xsl:for-each>
  <xsl:for-each
  select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dcterms:alt
  ernative/rdf:Bag/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dcterms',',','alternative')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
  </xsl:for-each>
  <xsl:for-each
  select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:creator/r
  df:Seq/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc',',','creator')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
  </xsl:for-each>
  <xsl:for-each
  select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:subject/r
  df:Bag/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc',',','subject')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
  </xsl:for-each>

```

```

<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:descript
ion/dc2ap:problem/*">
<field>
<xsl:attribute name="name">
<xsl:value-of select="concat('dc2ap',!, 'problem')"/>
</xsl:attribute>
<xsl:value-of select="text()"/>
</field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:descript
ion/dc2ap:motivation/rdf:Bag/*">
<field>
<xsl:attribute name="name">
<xsl:value-of select="concat('dc2ap',!, 'motivation')"/>
</xsl:attribute>
<xsl:value-of select="text()"/>
</field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:descript
ion/dc2ap:example/rdf:Bag/*">
<field>
<xsl:attribute name="name">
<xsl:value-of select="concat('dc2ap',!, 'example')"/>
</xsl:attribute>
<xsl:value-of select="text()"/>
</field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:descript
ion/dc2ap:knownUses/rdf:Bag/*">
<field>
<xsl:attribute name="name">
<xsl:value-of select="concat('dc2ap',!, 'knownUses')"/>
</xsl:attribute>
<xsl:value-of select="text()"/>
</field>
</xsl:for-each>

```

```

<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:descript
ion/dc2ap:context/*">
<field>
<xsl:attribute name="name">
<xsl:value-of select="concat('dc2ap', '.', 'context')"/>
</xsl:attribute>
<xsl:value-of select="text()"/>
</field>
</xsl:for-each>

```

```

<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:publishe
r/rdf:Seq/*">
<field>
<xsl:attribute name="name">
<xsl:value-of select="concat('dc', '.', 'publisher')"/>
</xsl:attribute>
<xsl:value-of select="text()"/>
</field>
</xsl:for-each>

```

```

<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:contribu
tor/rdf:Seq/*">
<field>
<xsl:attribute name="name">
<xsl:value-of select="concat('dc', '.', 'contributor')"/>
</xsl:attribute>
<xsl:value-of select="text()"/>
</field>
</xsl:for-each>

```

```

<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:date/dct
erms:created/dcterms:W3CDTF/*">
<field>
<xsl:attribute name="name">
<xsl:value-of select="concat('dcterms', '.', 'created')"/>
</xsl:attribute>
<xsl:value-of select="text()"/>
</field>
</xsl:for-each>

```

```

<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:date/dct
erms:modified/dcterms:W3CDTF/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dcterms',',', 'modified')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:type/dc
2aptp:analysisPattern/rdfs:label">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc2aptp',',', 'rdfs',',', 'label')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:type/dc
2aptp:analysisPattern/rdaelem:formOfNotation">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc2aptp',',', 'rdaelem',',', 'formOfNotation')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:format/r
df:Bag/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc',',', 'format')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:source/*
">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc',',', 'source')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>

```

```

<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:language/dcterms:ISO639-3/rdf:value/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('language',',', 'value')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:language/dcterms:ISO639-3/rdfs:label/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('language',',', 'label')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:relation/dcterms:isVersionOf/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dcterms',',', 'isVersionOf')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:relation/dcterms:isReplacedBy/rdf:Alt/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dcterms',',', 'isReplacedBy')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:relation/dcterms:replaces/rdf:Alt/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dcterms',',', 'replaces')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>

```

```

<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:relation/
dcterms:isPartOf/rdf:Bag/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dcterms',',', 'isPartOf')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:relation/
dcterms:hasPart/rdf:Bag/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dcterms',',', 'hasPart')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:relation/
rdaelem:note/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('rdaelem',',', 'note')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:relation/
dc2ap:isDesignedWith/rdf:Bag/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc2ap',',', 'isDesignedWith')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:relation/
dc2ap:shouldAvoid/rdf:Bag/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc2ap',',', 'shouldAvoid')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>

```

```

<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:relation/
dc2ap:complementedBy/rdf:Bag/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc2ap',',', 'complementedBy')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:coverag
e/rdf:Bag/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc',',', 'coverage')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc:rights/rd
f:Bag/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc',',', 'rights')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc2ap:histo
ry/rdf:Seq/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc2ap',',', 'history')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc2ap:requi
rements/dc2ap:functionalRequirements/rdf:Seq/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc2ap',',', 'functionalRequirements')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>

```



```

<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc2ap:requ
irements/dc2ap:nonFunctionalRequirements/rdf:Seq/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc2ap',',', 'nonFunctionalRequirements')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc2ap:requ
irements/dc2ap:dependenciesAndContributions/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc2ap',',', 'dependenciesAndContributions')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc2ap:requ
irements/dc2ap:dependencyGraph/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc2ap',',', 'dependencyGraph')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc2ap:requ
irements/dc2ap:contributionGraph/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc2ap',',', 'contributionGraph')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc2ap:requ
irements/dc2ap:conflictAndResolution/rdf:Bag/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc2ap',',', 'conflictAndResolution')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>

```

```

<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc2ap:requ
irements/dc2ap:priorities/*">
<field>
<xsl:attribute name="name">
<xsl:value-of select="concat('dc2ap',',', 'priorities')"/>
</xsl:attribute>
<xsl:value-of select="text()"/>
</field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc2ap:requ
irements/dc2ap:participants/rdf:Bag/*">
<field>
<xsl:attribute name="name">
<xsl:value-of select="concat('dc2ap',',', 'participants')"/>
</xsl:attribute>
<xsl:value-of select="text()"/>
</field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc2ap:mod
elling/dc2ap:behaviour/dc2ap:useCaseDiagram/*">
<field>
<xsl:attribute name="name">
<xsl:value-of select="concat('dc2ap',',', 'useCaseDiagram')"/>
</xsl:attribute>
<xsl:value-of select="text()"/>
</field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc2ap:mod
elling/dc2ap:behaviour/dc2ap:collaborationSequenceDiagrams/rdf:Bag/*">
<field>
<xsl:attribute name="name">
<xsl:value-of select="concat('dc2ap',',', 'collaborationSequenceDiagrams')"/>
</xsl:attribute>
<xsl:value-of select="text()"/>
</field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc2ap:mod
elling/dc2ap:behaviour/dc2ap:activityStateDiagrams/rdf:Bag/*">
<field>
<xsl:attribute name="name">
<xsl:value-of select="concat('dc2ap',',', 'activityStateDiagrams')"/>
</xsl:attribute>
<xsl:value-of select="text()"/>
</field>
</xsl:for-each>

```

```

<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc2ap:mod
elling/dc2ap:structure/dc2ap:classDiagram/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc2ap',',', 'classDiagram')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc2ap:mod
elling/dc2ap:structure/dc2ap:classDescriptions/rdf:Bag/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc2ap',',', 'classDescriptions')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc2ap:mod
elling/dc2ap:structure/dc2ap:relationshipDescriptions/rdf:Bag/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc2ap',',', 'relationshipDescriptions')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc2ap:mod
elling/dc2ap:solutionVariants/rdf:Bag/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc2ap',',', 'solutionVariants')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>
<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc2ap:resul
tingContext/rdf:Bag/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc2ap',',', 'resultingContext')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>

```

```

<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc2ap:designGuidelines/rdf:Bag/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc2ap',',', 'designGuidelines')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>

<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc2ap:consequences/dc2ap:positive/rdf:Bag/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc2ap',',', 'positive')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>

<xsl:for-each
select="foxml:datastream/foxml:datastreamVersion[last()]/foxml:xmlContent/rdf:RDF/dc2ap:consequences/dc2ap:negative/rdf:Bag/*">
  <field>
  <xsl:attribute name="name">
  <xsl:value-of select="concat('dc2ap',',', 'negative')"/>
  </xsl:attribute>
  <xsl:value-of select="text()"/>
  </field>
</xsl:for-each>

</doc>
</add>
</xsl:template>

<xsl:template match="/foxml:digitalObject" mode="inactiveFedoraObject">
<delete>
<id><xsl:value-of select="$PID"/></id>
</delete>
</xsl:template>

</xsl:stylesheet>

```