ANGÉLICA APARECIDA DE ALMEIDA RIBEIRO

APRIMORA: UMA INFRAESTRUTURA SEMÂNTICA PARA REUSO DE ARTEFATOS COMPUTACIONAIS

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae.*

VIÇOSA
MINAS GERAIS – BRASIL
2014

Ficha catalográfica preparada pela Biblioteca Central da Universidade Federal de Viçosa - Câmpus Viçosa

T
R484a
2014

Ribeiro, Angélica Aparecida de Almeida, 1990-
APRImora : uma infraestrutura semântica para reuso de artefatos computacionais / Angélica Aparecida de Almeida Ribeiro. – Viçosa, MG, 2014.
ix, 89f. : il. (algumas color.) ; 29 cm.

Inclui apêndices.
Orientador: Jugurta Lisboa Filho.
Dissertação (mestrado) - Universidade Federal de Viçosa.
Referências bibliográficas: f.87-89.

1. Programação. 2. Web semântica. 3. Sistemas de Recuperação da informação. 4. Projeto. I. Universidade Federal de Viçosa. Departamento de Informática. Programa de Pós-graduação em Ciência da Computação. II. Título.

CDD 22. ed. 005.711

“Mesmo que eu tivesse o dom da profecia, e conhecesse todos os mistérios e toda a ciência, mesmo que tivesse toda a fé, a ponto de transportar montanhas, se não tiver caridade, não sou nada” (I Coríntios 13, 2-3)

AGRADECIMENTOS

Hoje subo mais um degrau na minha vida, sustentada pelas mãos de Deus e apoiada pelas orações daqueles que me amam. Mais uma vitória que não consegui sozinha, afinal sozinho ninguém é capaz de nada. Por isso tenho muito que agradecer a todos que trilharam comigo esse caminho, e se por ventura vier a ser injusta com alguém e esquecer de agradecer peço que me desculpem e sintam-se agradecidos.

Em primeiro lugar gostaria de agradecer a Deus pelo dom da minha vida, por estar comigo em cada passo que dou, me impulsionando quando penso em desistir, me dando forças quando sinto que elas se esgotaram, enfim, por me lembrar sempre "que até aqui sua mão me sustentou". Deus não escolhe os capacitados, mas realmente capacita seus escolhidos, por isso Deus muito obrigada, te amo, te adoro e te bendigo por tudo que fez em minha vida e por tudo aquilo que ainda irá fazer.

Aos meus pais, Vianelo e Eunice, que são meu exemplo, foram meus primeiros mestres, me ensinaram a ser quem sou hoje. O apoio de vocês em cada passo que dou é o que me motiva a seguir em frente e sempre querer o melhor. Essa vitória também é de vocês. Muito obrigada por toda dedicação, por todas as palavras de conforto e incentivo quando precisei, muito obrigada também pelas broncas quando mereci. Pelo ombro amigo quando precisei chorar, mas muito obrigada por me ensinarem a sorrir novamente. Muito obrigada pelo amor incondicional, e pelo abraço carinhoso e de acolhida todas as vezes que chego em casa. Pai e mãe amo muito vocês.

Ao meu irmão Clemilson a quem tenho um grande amor e admiração, obrigada por você existir. Ao meu irmão Geovani (*in memoriam*) o qual sinto tanta saudade, mas sei que lá pertinho de Deus tem guiado meus passos e festejado comigo todas as minhas vitórias.

Ao meu grande amor, meu noivo Marcos Tadeu, pelo amor, companheirismo, por sempre me dizer que eu sou capaz, por acreditar em mim, mesmo quando nem eu mesma acredito, por estar sempre do meu lado, por ter as palavras certas no momento certo. Viu meu amor, eu consegui! E não conseguiria sem você. Dedico a você essa vitória.

Ao professor Jugurta, pela paciência e sabedoria com que me orientou e auxiliou na condução desse trabalho. Saiba que levarei comigo pelo resto da vida tudo que me ensinou, e serei eternamente grata por tudo que fez por mim. Mas não te levarei apenas como orientador, mas sim como um amigo.

Aos professores Alcione e Lucas Vegi que foram essenciais no desenvolvimento desse trabalho.

Ao amigo Sergio Murilo Stempliuc, a quem me apresentou o caminho da pesquisa, através da iniciação científica, se hoje posso levantar o troféu de mais essa vitória é graças a você que sempre acreditou em mim.

Ao amigo Marcelo Daibert que sempre torceu pelo meu sucesso e sempre esteve disposto a me ajudar, sua amizade e seus ensinamentos vou sempre levar comigo.

Ao Altino, secretário da pós graduação, pelo carinho e por estar sempre disposto a ajudar. Que Deus lhe pague por toda a sua dedicação.

A todos os professores que passaram pela minha vida, aos quais foram de grande importância no meu desenvolvimento tanto como pessoa quanto como profissional.

Aos amigos e amigas do DPI, pelos momentos de conversa e descontração, que me davam forças e não me deixavam desanimar.

A todos os mestres que já passaram pela minha vida.

Por todos aqueles que torcem pela minha vitória.

Enfim, agradeço a cada pessoa que de alguma forma com apenas um sorriso, uma palavra amiga, ou simplesmente por existir, me ajudaram a perceber que sou capaz e torceram pela minha vitória.

Muito Obrigada!

SUMÁRIO

LISTA DE FIGURAS

Artigo I

Artigo 2

LISTA DE TABELAS

RESUMO

RIBEIRO, Angélica Aparecida de Almeida, M.Sc., Universidade Federal de Viçosa, junho de 2014. **APRImora: Uma Infraestrutura Semântica para Reuso de Artefatos Computacionais.** Orientador: Jugurta Lisboa Filho. Coorientador: Alcione de Paiva Oliveira.

A Infraestrutura de Reuso de Padrões de Análise (APRI) foi proposta por Vegi (2012) para possibilitar a disseminação e evolução de padrões de análise usando uma abordagem voltada a Web Services. Os padrões de análise são armazenados em um repositório e documentados utilizando um perfil de metadados específicos denominado Perfil de Aplicação do Dublin Core para Padrões de Análise (DC2AP). Atualmente, a APRI é composta apenas por padrões de análise, sendo assim, essa dissertação propõe estender a estrutura da APRI para uma infraestrutura que seja capaz de suportar padrões de projeto. Padrões de projeto identificam os principais aspectos de uma estrutura de projeto que seja comum e capaz de ser útil para a criação de outros projetos orientados a objetos, descrevendo como, onde e em qual situação eles devem ser aplicados e quais as consequências de seu uso. Com a adição de padrões de projeto na APRI será possível auxiliar na catalogação e busca desses padrões, facilitando a sua descoberta, estudo e o reuso mais amplo do mesmo. Dentre os objetivos específicos está o desenvolvimento de um perfil de metadados para documentar padrões de projeto, denominado Perfil de Aplicação Dublin Core para documentar Padrões de Projeto (DC2DP). Mas não basta apenas adicionar repositórios à estrutura da APRI, é necessário dar subsídios para que os padrões possam ser recuperados e dessa forma reutilizados. Sendo assim, essa dissertação propõe estender a APRI para uma infraestrutura semântica para reuso de artefatos computacionais, que recebe o nome de APRImora. Dessa forma pretende-se melhorar o potencial de reuso dos padrões presentes nessa infraestrutura, através da utilização das camadas da Web Semântica para adicionar semântica aos elementos e apoiar na recuperação de forma precisa dos padrões. Esta arquitetura irá auxiliar os projetistas a encontrar de forma mais eficiente os padrões existentes, levando-os a adicionar estes padrões em seus projetos, poupando assim tempo no desenvolvimento sem a necessidade de iniciar o projeto do zero.

ABSTRACT

RIBEIRO, Angélica Aparecida de Almeida, M.Sc., Universidade Federal de Viçosa, June 2014. **APRImora: A Semantic Infrastructure for Computational Artifact Reuse.** Adviser: Jugurta Lisboa Filho. Co-Adviser: Alcione de Paiva Oliveira.

The Analysis Pattern Reuse Infraestructure (APRI) was proposed by Vegi (2012) to enable the dissemination and evolution of analysis patterns using a focused approach aimed at Web Services. The analysis patterns are stored in a repository and documented using a specific metadata profile called Dublin Core Application Profile to Analysis Patterns (DC2AP).Currently, the APRI is composed only by Analysis Patterns, thus this thesis proposes an extension of the APRI structure to an infrastructure that can be capable of supporting design patterns. Design Patterns are patterns that identify the primary aspects of a project structure that are common and capable of being useful in the creation of other object oriented projects, describe ng how, where and in which situations they must be applied and are the consequences of its use. With the addition of design patterns in the APRI it will be possible to help in the cataloging and search of this patterns, facilitating their detection, study and a wider reuse of it. Among the specific objectives there is the development of a metadata profile to document design patterns, called Dublin Core Application Profile to Design Patterns (DC2DP).But just adding more repositories to the APRI structure is not enough, it is necessary to give subsidies so that the standards can be retrieved and in this way reused. Thus, this thesis also proposes the APRI extension to a Semantic Infrastructure for reuse of computational artifacts, that receives the name of APRImora. Thus it is intended to improve the potential of reusing patterns that are on this infrastructure, through the use of Semantic Web layers to add semantic elements and support the retrieving of patterns in a more precise way. This architecture will help the designers to find in the most efficient way the existing patterns, leading them to add this patterns in your projects, thus saving time without the need of starting development from scratch.

1 INTRODUÇÃO

1.1 O problema e a sua importância

O reuso de Padrões foi proposto originalmente na área de arquitetura (ALEXANDER et al., 1977) e posteriormente adaptado para a área de Engenharia de Software. A importância da utilização de padrões se dá por serem soluções já utilizadas anteriormente, que deram certo e, portanto, podem auxiliar no desenvolvimento de novos projetos sem precisar desenvolver tudo do zero.

Na área de Engenharia de Software, padrões de projeto (GAMMA et al., 1995), padrões de análise (FOWLER, 1997), frameworks e componentes são exemplos de artefatos computacionais reutilizáveis durante o processo de desenvolvimento de software. Portanto, para que esses artefatos possam ser recuperados e reutilizados, uma solução seria armazená-los em um repositório de padrões onde eles pudessem ser facilmente acessados e reutilizados. Além disso, os padrões armazenados no repositório precisam ser documentados com base em um conjunto de elementos comuns para facilitar assim a sua recuperação.

Segundo Vegi (2012), os padrões de análise em sua maioria encontram-se em livros e artigos científicos, documentados com poucos detalhes, o que compromete a sua reutilização. Além disso, Hümmelgen (1999) afirma que a maneira de se descrever padrões difere de autor para autor, e não existe modelo único para a descrição de um padrão. Muitos autores se inspiram no estilo de Alexander et al. (1977), outros seguem o modelo apresentado por Gamma et al. (1995), e outros ainda criaram seu próprio meio de descrever suas ideias, o que dificulta a divulgação e consequentemente a reutilização desses padrões.

A não existência de um padrão de metadados para documentar os padrões dificulta a busca e recuperação desses. Dessa forma, o primeiro passo para a recuperação de informações no repositório que armazena padrões é a definição de um conjunto de metadados que padronize a descrição dos padrões.

Baseado nisso, Vegi (2012) propôs uma Infraestrutura de Reuso de Padrões de Análise (APRI), composta por um repositório de padrões de análise, documentados com base em um perfil de aplicação Dublin Core específico para documentar Padrões de Análise, denominado DC2AP (*Dublin Core Application Profile to Analysis Patterns*). Uma APRI tem como objetivo possibilitar a disseminação e evolução de padrões de análise usando uma abordagem voltada a Web Services (VEGI et al., 2012).

Inicialmente a estrutura da APRI era composta apenas por padrões de análise, porém Vegi (2012) previa a extensão dessa arquitetura adicionando a ela outros artefatos de software como, por exemplo, padrões de projeto, padrões de arquitetura, frameworks e componentes. Essa dissertação irá discorrer sobre os padrões de projeto que, assim como os padrões de análise, em sua maioria, encontram-se em livros e artigos científicos. Sendo possível encontrá-los na Internet, porém documentados com poucos detalhes, comprometendo a sua reutilização.

Um dos objetivos dessa dissertação é apresentar a extensão da estrutura da APRI adicionando a ela um repositório de padrões de projeto, enriquecendo assim sua estrutura e auxiliando ainda mais os projetistas, uma vez que os padrões de projeto e os de análise estarão se comunicando, facilitando assim a busca dos projetistas e a reutilização desses dois tipos de artefatos. Os padrões de projeto armazenados na estrutura da APRI necessitam ser documentados utilizando padrão (*standard*) de metadados específico para documentar este tipo de artefato.

Segundo Gamma et al. (1995) um padrão de projeto identifica os principais aspectos de uma estrutura de projeto que seja comum e capaz de ser útil para a criação de outros projetos orientados a objetos. Assim sendo, a importância da adição de padrões de projeto em uma APRI se dá de acordo com o que diz Booch (2000, apud GAMMA et al., 2000, p. 9) pela utilização de padrões durante o desenvolvimento de um sistema ser capaz de produzir arquiteturas menores, mais simples e muito mais compreensíveis do que em arquiteturas onde estes padrões são ignorados.

Mas não basta apenas adicionar os padrões em um repositório, é necessário dar subsídios para que estes possam ser recuperados e dessa forma reutilizados. Os padrões armazenados no repositório da APRI podem ser recuperados a partir de um navegador Web, porém, por utilizar máquinas de busca, o sistema pode retornar uma quantidade de resultados irrelevantes para o contexto da pesquisa e inadequados para resolver o propósito da busca.

Segundo Berners-Lee et al. (2001) a Web se desenvolveu mais rapidamente como um meio de documentos para as pessoas, em vez de se desenvolver para os dados e informações que podem ser processados automaticamente. Como consequência deste fato, ao realizar uma pesquisa, na maioria das vezes é obtido como resultado milhares de ocorrências com pouca ou nenhuma relevância. Isso ocorre devido ao significado do conteúdo existente na Web não ser

processável por máquina, além disso a capacidade dos softwares é limitada no que diz respeito à interpretação de frases e extração de informações úteis para os usuários (ANTONIOU et al., 2008).

Diante disso surgiu a Web Semântica que tem por objetivo compensar esse problema através do fornecimento de meios para organizar os dados e permitir que esses possam ser interpretados por computadores.

Portanto, este trabalho propõe estender a APRI para uma arquitetura que tenha como base a Web Semântica, denominada APRImora, que seja capaz de auxiliar o usuário na recuperação da informação contida em seus repositórios de forma precisa.

1.2 Objetivos

O objetivo geral desta pesquisa é adicionar à estrutura da APRI componentes semânticos, dando origem assim à APRImora uma infraestrutura de reuso semântico para artefatos computacionais que, apoiada em tecnologias da Web Semântica, auxilie na recuperação precisa de artefatos computacionais.

Especificamente, pretende-se:

a) Definir um perfil do padrão de metadados Dublin Core, específico para descrição de padrões de projeto (DC2DP);
b) Adicionar à estrutura da APRI um repositório de padrões de projeto;
c) Propor um editor de metadados do perfil DC2DP para auxiliar na documentação dos padrões de projeto;
d) Estender a APRI adicionando a ela tecnologias da Web Semântica para que se torne uma infraestrutura semântica de reuso de artefatos computacionais.

1.3 Metodologia

Este trabalho teve como base o trabalho de Vegi (2012), onde o autor define um padrão para documentar padrões de análise. Baseado nisso, esse trabalho procurou entender se existia a necessidade de criação de um novo padrão para documentar padrões de projeto ou se seria suficiente apenas a criação de um perfil a partir do padrão proposto por Vegi para documentar padrões de análise. O trabalho de Monteiro (2013) também foi de grande importância para essa dissertação, uma vez que em seu trabalho é proposto uma arquitetura baseada em Web Semântica para repositórios digitais educacionais na área de saúde. Monteiro

apresenta como as camadas da Web Semântica auxiliam na recuperação de objetos de aprendizado na área de saúde de forma refinada.

Foram realizadas algumas ações com o intuito de alcançar os objetivos apresentados neste trabalho, as quais são apresentadas a seguir:

1. Investigar trabalhos relacionados: inicialmente foi realizada a revisão da dissertação de Vegi (2012), que apresenta a proposta de um perfil para documentação de padrões de análise a fim de facilitar e auxiliar o projetista no reuso de ideias já comprovadas anteriormente no que diz respeito aos padrões de análise, armazenando-os em uma infraestrutura de reuso, chamada APRI. Posteriormente, uma investigação sobre padrões de projeto foi realizada, entre elas a leitura do livro: "*Design Patterns: Elements of Reusable Object-Oriented Software*" (GAMMA et al., 1995), onde os autores apresentam um *template* para documentar padrões de projeto, este *template* foi de grande importância para a realização dessa dissertação. Em seguida a tese de Monteiro (2013) foi estudada, e com o estudo desse trabalho foi possível conhecer a importância da adição das camadas da Web Semântica em repositórios e o auxílio destas na obtenção de resultados refinados nas pesquisas realizadas.

2. Realizar a comparação do perfil de metadados Dublin Core (DC) e o DC2AP proposto por Vegi (2012) com o *template* de Gamma et al. (1995): esta etapa foi realizada para verificar se existia a necessidade de criação de um novo perfil de metadados para documentar padrões de projeto, ou se os elementos do DC2AP utilizados para documentar padrões de análise eram também suficientes para documentar padrões de projeto. Essa verificação foi realizada em duas etapas, descritas com mais detalhes a seguir:

- Primeiro foi realizada uma comparação dos elementos do perfil Dublin Core com o *template* proposto por Gamma et al. (1995). Essa comparação foi feita com o intuito de encontrar os elementos semelhantes e os diferentes existentes entre os dois. Após a comparação, foram detectados elementos semelhantes entre os dois, além de elementos importantes existentes no *template*, porém não encontrados no perfil DC, sendo assim esses elementos foram incorporados ao perfil DC, gerando um novo perfil.
- Em seguida, os elementos existentes no perfil criado no primeiro passo foram comparados aos elementos do DC2AP. Avaliando essa comparação foi possível definir a necessidade de um novo perfil para a documentação de padrões de projeto,

embora alguns elementos fossem semelhantes, nem todos os elementos existentes no perfil do DC criado possuíam correspondentes no DC2AP, além disso, o padrão de análise compreende a fase de planejamento do software, sua documentação apresenta diagramas, entre outros elementos que caracterizam essa fase. Já o padrão de projeto compreende a fase de desenvolvimento, sendo de extrema importância a documentação dos códigos fonte, entre outros elementos. Essa diferença levou a criação de um novo perfil.

3. Criação de um novo perfil: após a análise apresentada no passo 2, foi proposto o DC2DP (*Dublin Core Application Profile to Design Patterns*). Nessa etapa foi feita a descrição técnica dos elementos, essa descrição pode ser encontrada no Apêndice A dessa dissertação.

4. Estudo da estrutura da APRI e adição de um repositório para documentar padrões de projeto: o objetivo de uma Infraestrutura de Reuso para Padrões de Análise (APRI) é prover ambientes que propiciem a disseminação e evolução de padrões de análise usando uma abordagem voltada a *Web Services*. Além disso, uma das suas principais funções é permitir que os padrões possam ser descobertos pelos usuários, fornecendo ferramentas que possibilitem sua recuperação e consequentemente a reutilização dos padrões de análise. Assim sendo, a proposta inicial da APRI só abrange os padrões de análise. Esta dissertação, portanto, adiciona a APRI um repositório de padrões de projeto documentados utilizando o Perfil de Aplicação Dublin Core para documentar Padrões de Projeto (DC2DP).

5. Documentar padrões de projeto: nessa etapa os padrões de projeto apresentados no trabalho de Gamma et al. (1995) foram documentados utilizando o perfil de metadados DC2DP e armazenados no repositório para padrões de projeto adicionado na APRI.

6. Editor de metadados para documentar padrões de projeto: para que os padrões de projeto sejam documentados e armazenados nos repositórios da APRI, foi desenvolvido, juntamente com um aluno de Iniciação Científica, um Editor de Metadados DC2DP. Atualmente esse editor tem como funcionalidades a edição dos metadados de acordo com os elementos existentes no DC2DP, a geração de RDF do padrão documentado e a validação dos elementos.

7. Adicionando semântica à APRI: essa etapa foi realizada com base no estudo da Web Semântica como auxilio na recuperação de informação de forma eficaz e precisa, além

da investigação de serviços Web semânticos que façam uso dos padrões presentes no repositório e descritos utilizando *Linked Data*. Com a adição de semântica à APRI é esperado que o potencial de reuso dos padrões presentes nessa infraestrutura seja melhorado.

1.4 Organização da dissertação

Esta dissertação foi elaborada de acordo com um dos formatos recomendados pela Comissão do Programa de Pós-Graduação em Ciência da Computação da UFV. E está organizada como uma coletânea de artigos produzidos durante a pesquisa. São dois artigos resultantes desse trabalho. Sendo assim, a dissertação está organizada como se segue:

O Capítulo 1 apresenta a introdução onde trata o problema e a importância do trabalho realizado, e apresenta os objetivos a serem alcançados com essa pesquisa e a metodologia.

O Capítulo 2 é composto dos artigos resultantes da pesquisa realizada. O Artigo I (Seção 2.1) apresenta a proposta do Perfil de Aplicação Dublin Core para Padrões de Projeto (DC2DP). O Artigo II (Seção 2.2) apresenta a proposta da extensão da APRI com a adição de semântica em sua estrutura, dando origem assim à APRImora.

O Capítulo 3 apresenta as conclusões gerais e os resultados obtidos. Ainda nesse capítulo são sugeridos trabalhos futuros para o aprimoramento da pesquisa apresentada nesta dissertação.

O Apêndice A inclui uma descrição semântica dos elementos presentes no perfil de metadados proposto nesta dissertação.

O Apêndice B inclui uma descrição técnica detalhada do perfil de metadados proposto nesta dissertação, onde pode ser encontrada a descrição semântica completa dos elementos do perfil de metadados e detalhes das regras de aplicação, sintaxe e vocabulários controlados associados a cada um desses elementos.

As referências completas dos artigos que compõem esta dissertação são apresentadas a seguir:

- RIBEIRO, A. A. A.; LISBOA FILHO, J.; VEGI, L., F., M., OLIVEIRA, A. P. **DC2DP: a Dublin Core Application Profile to Design Patterns.** *In: INTERNATIONAL CONFERENCE ON ENTERPRISE INFORMATION SYSTEMS*

(ICEIS), 16, 2014, Lisboa, Portugal. Proceedings… Lisboa: SCITEPRESS, 2014, v.2, p. 209-215.

- RIBEIRO, A. A. A.; LISBOA FILHO, J.; VEGI, L. F. M.; OLIVEIRA, A. P.; FONSECA, E. J. de S. **APRImora: Uma Arquitetura Baseado em Web Semântica para Recuperação de Informações em Repositórios de Padrões.** A ser submetido posteriormente a uma revista.

2 ARTIGOS

Este capítulo apresenta uma coletânea contendo 2 artigos resultantes da pesquisa que deu origem a esta dissertação. O primeiro artigo, intitulado "*DC2DP: a Dublin Core Application Profile to Design Patterns*", apresenta a proposta do Perfil de Aplicação Dublin Core para documentar Padrões de Projeto. O segundo artigo, intitulado "APRImora: Uma Arquitetura Baseado em Web Semântica para Recuperação de Informações em Repositórios de Padrões", apresenta a extensão da APRI com a adição de semântica em sua estrutura, dando origem assim à APRImora.

2.1 Artigo I: DC2DP: a Dublin Core Application Profile to Design Patterns

Angélica Aparecida de Almeida Ribeiro, Jugurta Lisboa Filho,
Lucas Francisco da Matta Vegi e Alcione de Paiva Oliveira

In: INTERNATIONAL CONFERENCE ON ENTERPRISE INFORMATION SYSTEMS (ICEIS), 16, 2014, Lisboa, Portugal. Proceedings… Lisboa: SCITEPRESS, 2014, v.2, p. 209-215

RESUMO

Padrões de projeto descrevem soluções reutilizáveis para problemas existentes no desenvolvimento de software orientado a objetos. Os padrões de projeto em sua maioria são documentados de forma textual em livros e artigos científicos, o que dificulta o processamento por computador, sua difusão e uma reutilização mais ampla. Também é possível encontrar esses padrões na Internet, porém documentados com poucos detalhes, dificultando sua compreensão e consequentemente sua reutilização. Esse artigo apresenta um perfil de aplicação do padrão de metadados Dublin Core específico para padrões de projeto, denominado DC2DP. O objetivo é permitir que os padrões de projeto sejam documentados de modo a fornecer ao usuário uma descrição mais detalhada e padronizada, além de possibilitar o processamento automático por meio de serviços web. O artigo também estende uma Infraestrutura de Reuso de Padrões de Análise (APRI), adicionando a esta um repositório para padrões de projeto, possibilitando assim a catalogação e busca desses padrões, facilitando a sua descoberta, estudo e reuso.

Palavras-chave: Padrão de Projeto, Dublin Core, Reuso, *Web Services*

ABSTRACT

Design patterns describe reusable solutions to existing problems in object-oriented software development. Design patterns are mostly documented in written form in books and scientific papers, which hinders processing them via computer, their diffusion, and their broader reuse. They can also be found on the internet, though documented with little detail, which makes it hard to understand and consequently reuse them. This paper presents an application profile of the Dublin Core metadata standard specific for design patterns, called DC2DP. The goal is to allow design patterns to be documented so as to provide the user with a more detailed and standardized description, besides enabling automatic processing through web services. The paper also extends an Analysis Patterns Reuse Infrastructure (APRI) by adding a design

pattern repository to it, thus allowing these patterns to be cataloged and searched, which makes their discovery, study, and reuse easier.

Keywords: Design Pattern, Dublin Core Standard, Reuse, Web Services

1 INTRODUCTION

In order to locate a book in a library, a work of art in a museum, or a city map in a map repository, the various catalogs of these objects need to be consulted. In the digital era, this type of catalog corresponds to a metadata (data on data) repository containing the description and information on how to obtain or locate the documented object. Metadata must follow a standardized documentation structure (element set) so that the systems are able to achieve interoperability in its search modules. The Dublin Core Metadata Element Set (DCMI, 1998) was defined as a way to serve several areas, making available a minimum set of mandatory elements in documenting any type of object. Moreover, an application profile may be defined, i.e., a standard customization for a specific area.

In software engineering, design patterns (GAMMA et al., 1995), analysis patterns (FOWLER, 1997), frameworks, and components are examples of reusable computational artifacts during software development. The issue is that these reusable artifacts are not easily found and most times programmers and designers choose to develop their solutions from scratch instead of researching the existence of previously tested solutions validated in other systems.

Thus, Vegi et al. (2012b) proposed an Analysis Patterns Reuse Infrastructure (APRI) made up of a repository of analysis patterns, documented based on a Dublin Core Application Profile specific for analysis patterns.

According to Gamma et al. (1995), a design pattern identifies the mains aspects of a design structure that is common and possibly useful for creating other object-oriented projects. Each design pattern is able to focus on one particular problem or topic of an object-oriented design. The pattern describes how, where, and in which situation it must be employed, and the consequences of its use. Gamma et al. (1995) states that the object-oriented architectures, when well structured, may carry several patterns. Gamma et al. also claims that one way of measuring the quality of an object-oriented system is to assess how the developers

used the common collaborations among its objects. The use of these patterns during the development of a system is able to produce smaller, simpler, and much more understandable architectures than in architectures in which these patterns are ignored.

Documenting these patterns helps to capture the design experience so that the designers can use them more effectively. For that end, these patterns must be documented and presented in some easily accessible and understandable catalog.

This paper proposes a specific Dublin Core Application Profile to document design patterns. The profile is based on elements of the DC2AP metadata profile, proposed by Vegi et al. (2012a) to document analysis patterns, and on the template used by Gamma et al. (1995). Moreover, this paper proposes extending the APRI structure by adding to it a repository of design patterns, thus allowing this type of pattern to be cataloged and reused.

The remaining of the paper is structured as follows. Section 2 reviews works related to design pattern catalogs, besides introducing DC2AP, a metadata profile to document analysis patterns. Section 3 introduces DC2DP, a Dublin Core Application Profile to document design patterns. Section 4 proposes extending the APRI structure, while section 5 presents the final considerations and proposes some future work.

2 RELATED WORKS

2.1 Design Pattern Catalogs

A design pattern catalog is made up of a set of related patterns with characteristics in common. These patterns may be used individually or be interconnected, since they may be used alongside each other. There are several design pattern catalogs, such as GoF Patterns (GAMMA et al., 1995), J2EE Patterns (ALUR et al., 2003), SOA Patterns (SOA PATTERNS, 2013), among others.

Each existing design pattern catalog uses a way of documenting the patterns that compose it, i.e., each one uses a set of elements to describe the pattern, with no standardized way of documenting design patterns.

2.2.1 GoF Pattern Catalog

Gamma et al. (1995) propose in their book a design pattern catalog, which became known as GoF Pattern Catalog, made up of 23 patterns classified according to two criteria: scope and purpose.

Regarding scope, the patterns may be split into class and objects. As for the purpose, the patterns are classified into creation, structural, and behavioral patterns.

In order to describe the design patterns in the catalog, the authors divided each pattern into sections according to the template proposed. The elements in the template used in documenting these patterns are: pattern name and classification, purpose and goal, also known as, motivation, applicability, structure, participants, collaborations, consequences, implementation, code examples, and known uses.

2.2.2 J2EE Pattern Catalog

Alur et al. (2003) presented in their book a design pattern catalog based on the work experience with the J2EE platform of Sun Java Center to clients worldwide.

The J2EE Pattern catalog is made up of 21 patterns split into presentation layer, business layer, and integration layer patterns. Each pattern is documented following a template. The elements in the template are: problem, forces, solution, consequences, and related patterns.

2.2.3 SOA Pattern Catalog

The design patterns in the catalog presented in SOA Patterns (2013) list the service-oriented principles when a dependency or relationship among services in an architecture must be highlighted.

The SOA Pattern catalog is made up of 83 patterns divided into the following categories: Service Implementation, Service Security, Service Contract Design, Legacy Encapsulation, Service Governance, Capability Composition, Service Messaging, Composition Implementation, Service Interaction Security, Transformation, REST-inspired.

The elements in the template to document the SOA patterns are: problem, solution, application, principles, related patterns, goals related to service-oriented computing.

2.2 Dublin Core Application Profile to Analysis Patterns

The Dublin Core Application Profile to Analysis Patterns (DC2AP) was proposed by Vegi et al. (2012a) to document analysis patterns. This metadata profile was created based on the template proposed by Pantoquilho et al. (2003) to document analysis patterns and on the Dublin Core metadata standard elements (DCMI, 1998).

According to Vegi et al. (2012a), the goal of DC2AP is to improve the recovery and reuse of analysis patterns by means of a description that allows the computer to perform a more precise treatment of the data previously not recovered by search engines. This task is performed based on detailed information provided on these patterns.

The DC2AP elements are associated to a Universal Resource Identifier (URI) and semantically described by a Resource Description Framework (RDF). Using URI and RDF enables using the Linked Data concept, which then allows the patterns to be made available in an environment where the data may be processed directly or indirectly by machines. The DC2AP elements are detailed in section 3.2.

3 DUBLIN CORE APPLICATION PROFILE TO DESIGN PATTERNS

This section describes the Dublin Core Application Profile to Design Patterns (DC2DP). In order to define the DC2DP profile elements, initially the design pattern documentation template elements proposed by Gamma et al. (1995) were compared with the elements of the DC2AP so as to analyze whether the same metadata profile could be used to document the analysis and design patterns.

Based on this comparison, a significant difference was found between the element sets since the analysis patterns are more targeted towards the phase of analysis of requirements and modeling of the problem, mainly focusing on documenting the functional and non-functional requirements, the class diagrams among other artifacts targeted towards the analysis phase of a piece of software. On the other hand, design patterns are targeted towards the solution phase of the design, in order to define object-oriented architectures, tending to document examples of source code for certain problems.

Thus, the Dublin Core Application Profile to Design Patterns (DC2DP) was specified based on the elements of the template proposed by Gamma et al. (1995) to specify design patterns and on the DC2AP elements.

DC2DP's main goal is to improve the recovery and reuse of design patterns, by providing more detailed information on these patterns, now in a recoverable by search engines format.

The next sub-sections detail the steps taken to define DC2DP. Sub-section 3.1 presents the mapping between the Dublin Core elements and those of the template proposed by Gamma et al. (1995). Sub-section 3.2 presents the mapping between the elements generated in the mapping of sub-section 3.1 and the elements of DC2AP. Finally, sub-section 3.3 presents the DC2DP elements with their application rules.

3.1 Mapping between the Elements of the Dublin Core Metadata Standard and Gamma's Template

Unlike the Dublin Core metadata standard, which is generic and, therefore, helps in documenting resources from several domains, the template proposed by Gamma et al. (1995) is specific for documenting design patterns, which is why it was chosen to be the basis for creating DC2DP.

Dublin Core	Gamma et al. (1995)
Title	Name Also Known As
Creator	No Equivalent
Subject	No Equivalent
Description	Intention Motivation Applicability Known Uses
Publisher	No Equivalent
Contributor	No Equivalent
Date	No Equivalent
Type	No Equivalent
Format	No Equivalent
Identifier	No Equivalent
Source	No Equivalent
Language	No Equivalent
Relation	Related Patterns
Coverage	No Equivalent
Rights	No Equivalent
No Equivalent	Classification
	Structure
	Participants
	Collaborations
	Consequences
	Implementation
	Sample Code

Figure 1. Mapping between the elements of the Dublin Core and the template by Gamma et al. (1995)

Hence, the first task in defining DC2DP was mapping the elements of the template of Gamma et al. (1995) and the Dublin Core elements. This mapping process was carried out by comparing all the elements of either structure, based on the elements' semantic correspondence. The result of this mapping can be seen in Figure 1.

3.2 Mapping between the DC2AP elements and Dublin Core profile with Gamma's Template elements

From the mapping described in sub-section 3.1, the elements from the Dublin Core profile and from the template by Gamma et al. (1995) were combined, thus originating the basic DC2DP structure.

Next, the elements of this structure were compared to the DC2AP elements, since there was a need to check whether these two patterns were very similar and whether a new Dublin Core profile had to be generated to document design patterns or just a profile based on DC2AP had to be created to document design patterns. The comparison was similar to the one made with Dublin Core, with the elements being compared according to their semantics. The result of this mapping can be seen in Figure 2.

As it can be seen, many elements of Dublin Core + Gamma to not match the DC2AP profile, thus a new standard for documenting design patterns was proposed, whose elements are described in sub-section 3.3.

3.3 DC2DP: Dublin Core Application Profile to Design Patterns

Figure 3 shows the elements belonging to the Dublin Core Application Profile to Design Patterns (DC2DP), generated based on the mappings presented in sub-sections 3.1 and 3.2.

In order to generate DC2DP, the main characteristics in the Dublin Core pattern elements were taken into account, thus allowing the design patterns to be documented based on a generic metadata standard so as to transmit to the user necessary information on the patterns, aiding them in making the right choice and appropriately using these patterns.

DC2AP	Dublin Core + Gamma et al.
1.Identifier	Identifier
2.Title	Title
2.1 Alternative Title	Also Known As
3.Creator	Creator
4.Subject	Subject
5. Description	Description
5.1 Problem	Intention
5.2 Motivation	Motivation
5.2.1 Example	Applicability
5.2.2 Known Uses	Known Uses
5.3 Context	No Equivalent
6. Publisher	Publisher
7. Contributor	Contributor
8. Date	Date
8.1 Created	No Equivalent
8.2 Modified	No Equivalent
9. Type	Type
9.1 Notation	No equivalent
10. Format	Format
11. Source	Source
12. Language	Language
13. Relation	Relation
13.1 Is Version Of	No Equivalent
13.2 Is Replaced By	No Equivalent
13.3 Replaces	No Equivalent
13.4 Is Part Of	No Equivalent
13.5 Has Part	No Equivalent
13.6 Is Designed With	No Equivalent
13.7 Should Avoid	No Equivalent
13.8 Complemented By	Related Patterns
13.9 About	No Equivalent
14. Coverage	Coverage
15. Rights	Rights
16. History	No Equivalent
16.1 Event Date	No Equivalent
16.2 Author	No Equivalent
16.3 Reason	No Equivalent
16.4 Changes	No Equivalent
17. Requirements	No Equivalent
17.1 Functional Requirements	No Equivalent
17.2 Non-functional Requirements	No Equivalent
17.3 Dependencies and Contributions	Colaborations
17.3.1 Dependency Graph	No Equivalent
17.3.2 Contribution Graph	No Equivalent
17.4 Conflict Identification & Guidance to Resolution	Implementation
17.5 Priorities Diagram	No Equivalent
17.6 Participants	Participants
18. Modelling	Structure
18.1 Behaviour	No Equivalent
18.1.1 Use Case Diagram	No Equivalent
18.1.2 Collaboration/Sequence Diagrams	Structure
18.1.3 Activity/State Diagrams	No Equivalent
18.2 Structure	Structure
18.2.1 Class Diagram	Structure
18.2.2 Class Description	Participants
18.2.3 Relationship Descriptions	Collaborations
18.3 Solution Variants	No Equivalent
19. Resulting Context	No Equivalent
20. Design Guidelines	Implementation
21. Consequences	Consequences
21.1 Positive	Consequences
21.2 Negative	Consequences
No Equivalent	Classification
	Sample Code

Figure 2. Mapping between DC2AP and the elements of the Dublin Core + Gamma et al. (1995).

Gamma et al. (1995) present the elements in a catalog for design pattern documentation that the user must read in order to understand whether the pattern they use or intend to use is the right one for the problem being tackled. Some of these elements are: Applicability (Examples), Consequences, Structure (Structural Modeling), Participants (Class Description), Collaborations (Description of Relationships), Source-code Example. Therefore, these elements were kept in DC2DP.

Other elements, however, had their names changed when integrating DC2DP, which are presented in parenthesis as shown above. This change was done because the semantic meanings of the elements were similar to some elements in DC2AP, thus keeping both elements was considered redundant. Moreover, since DC2AP and DC2DP are used to document patterns and store them in a reuse infrastructure, i.e., APRI, it was considered worth maintaining the same name of the elements used in the two profiles whenever possible.

The DC2AP profile contains elements to control versions of the documented patterns, besides other elements for sharing usage experience. According to Vegi et al. (2012a), these characteristics were incorporated into DC2AP so as to allow new and enhanced versions of the patterns to be proposed with the collaboration of their usage experience. In addition, all versions of these patterns may relate with one another, a resource which enables users to recover the version that best and most efficiently meets their needs. Based on the importance of version control and on the help it provides in the mapping in sub-section 3.2, DC2DP kept the following elements: History and Relation.

After the elements that make up DC2DP had been defined, rules were proposed regarding obligatoriness, occurrence, and type of value of each of the elements proposed. These rules are presented in Figure 3.

Due to space constraints, the semantic description of each element that makes up DC2DP, as well as the details of the application rules, are not presented in this paper, but a detailed specification of the profile can be found at <http://www.dpi.ufv.br/projetos/apri>.

<table>
<tr><th colspan="3">Elements of DC2DP</th></tr>
<tr><td colspan="3">1.Identifier [M][S][UNS]</td></tr>
<tr><td>2. Title [M][S][St]</td><td colspan="2">2.1 Alternative Title [O][Mu][St]</td></tr>
<tr><td colspan="3">3. Classification [O][S][St]</td></tr>
<tr><td colspan="3">4. Creator [M][Mu][St]</td></tr>
<tr><td colspan="3">5. Subject [M][Mu][St]</td></tr>
<tr><td rowspan="3">6. Description [M][S][N]</td><td colspan="2">6.1 Problem [M][S][St]</td></tr>
<tr><td rowspan="2">6.2 Motivation [M][Mu][St]</td><td>6.2.1Example [M][Mu][St]</td></tr>
<tr><td>6.2.2 Known Uses [M][Mu][St]</td></tr>
<tr><td rowspan="2">7. Date [M][S][N]</td><td colspan="2">7.1 Created [M][S][D]</td></tr>
<tr><td colspan="2">7.2 Modified [Cd][S][D]</td></tr>
<tr><td colspan="3">8. Publisher [O][Mu][St]</td></tr>
<tr><td colspan="3">9. Contributor [Cd][Mu][St]</td></tr>
<tr><td colspan="3">10. Type [M][S][US]</td></tr>
<tr><td colspan="3">11. Source [Cd][S][US]</td></tr>
<tr><td colspan="3">12. Language [M][S][US]</td></tr>
<tr><td rowspan="7">13. Relation [Cd][S][N]</td><td colspan="2">13.1 Is version of [Cd][S][UNS]</td></tr>
<tr><td colspan="2">13.2 Is Replaced By [Cd][Mu][UNS]</td></tr>
<tr><td colspan="2">13.3 Replaces [Cd][Mu][UNS]</td></tr>
<tr><td colspan="2">13.4 Is Part of [O][Mu][UNS]</td></tr>
<tr><td colspan="2">13.5 Has Part [O][Mu][UNS]</td></tr>
<tr><td colspan="2">13.6 Related Patterns [O][Mu][UNS]</td></tr>
<tr><td colspan="2">13.7 Is Analyzed With [O][Mu][UNS]</td></tr>
<tr><td colspan="3">14. Coverage [O][Mu][St]</td></tr>
<tr><td colspan="3">15. Rights [Cd][Mu][US]</td></tr>
<tr><td rowspan="4">16. History[M][Mu][N]</td><td colspan="2">16.1 Event Date [M][S][D]</td></tr>
<tr><td colspan="2">16.2 Author [M][Mu][St]</td></tr>
<tr><td colspan="2">16.3 Reason [M][S][St]</td></tr>
<tr><td colspan="2">16.4 Changes [Cd][S][St]</td></tr>
<tr><td>17. Behavior Modelling[O][S][N]</td><td colspan="2">17.1 Sequence Diagram [O][S][U]</td></tr>
<tr><td rowspan="3">18. Structure Modelling [M][S][N]</td><td colspan="2">18.1 Object Diagram [O][S][U]</td></tr>
<tr><td rowspan="2">18.2 Class Diagram [M][S][U]</td><td>18.2.1 Class Description [M][S][St]</td></tr>
<tr><td>18.2.2Relation Description [M][S][St]</td></tr>
<tr><td rowspan="3">19. Sample Code [M][S][N]</td><td colspan="2">19.1 Programming Language [M][Mu][St]</td></tr>
<tr><td colspan="2">19.2 Code [M][Mu][St]</td></tr>
<tr><td colspan="2">19.3 Code Description [M][Mu][St]</td></tr>
<tr><td colspan="3">20. Design Guidelines [O][Mu][St]</td></tr>
<tr><td rowspan="2">21. Consequences [M][S][N]</td><td colspan="2">21.1 Positive [M][Mu][St]</td></tr>
<tr><td colspan="2">21.2 Negative [O][Mu][St]</td></tr>
<tr><th colspan="3">Rules' Acronyms</th></tr>
<tr><td>Obligatoriness</td><td>Occurrence</td><td>Value Type</td></tr>
<tr><td>[M] Mandatory
[O] Optional
[Cd] Conditional</td><td>[S] Single
[Mu] Multiple</td><td>[St] String
[D] Date
[U] URI
[N] Null
[UNS] URI, number or string
[US] URI or string</td></tr>
</table>

Figure 3. Elements of the Dublin Core Application Profile to Design Patterns.

4 DESIGN PATTERN REPOSITORY IN APRI

The goal of the Analysis Patterns Reuse Infrastructure (APRI) is to provide an environment that enables the dissemination and evolution of analysis patterns using an approach targeted to Web Services (VEGI et al., 2012b).

An APRI maintains a repository of analysis patterns documented with the DC2AP metadata profile. Thus, one of the main roles of an APRI is to allow the patterns to be discovered by the users, besides offering tools that allow the patterns to be updated.

This paper extends the APRI structure (Figure 4) by adding a repository of design patterns documented by means of the DC2DP profile. Hence, the analysis patterns may be referenced by design patterns and vice-versa.

A catalog with the design patterns documented by using the DC2DP profile and stored in the APRI repository is available at: <http://www.dpi.ufv.br/ projetos/apri>.

The main advantage of the proposed repository, is that, all the design patterns are gathered in the same place, this way facilitating retrieval of the patterns and consequently their reuse. Moreover, the proposed metadata profile to document the design patterns use the concept of linked data, which allows a design pattern to create a link between all the other patterns, that can somehow help in its implementation, thereby making a more complete documentation of the pattern.

The use of the linked data together with the semantic web concept, allow the data that is going to be recovered in the researches, to be more precise and return only what the user really desire, without excess data.

However, the disadvantage of this repository is the need of the user to have access to the internet, so that the data can be retrieved and the patterns visualized.

In order to exemplify the application of the metadata profile proposed in this study, Figure 5 presents a design pattern documented in DC2DP. The design pattern specified is the Singleton, proposed by Gamma et al. (1995). Due to space issues, only the mandatory elements are shown in the description of this pattern.

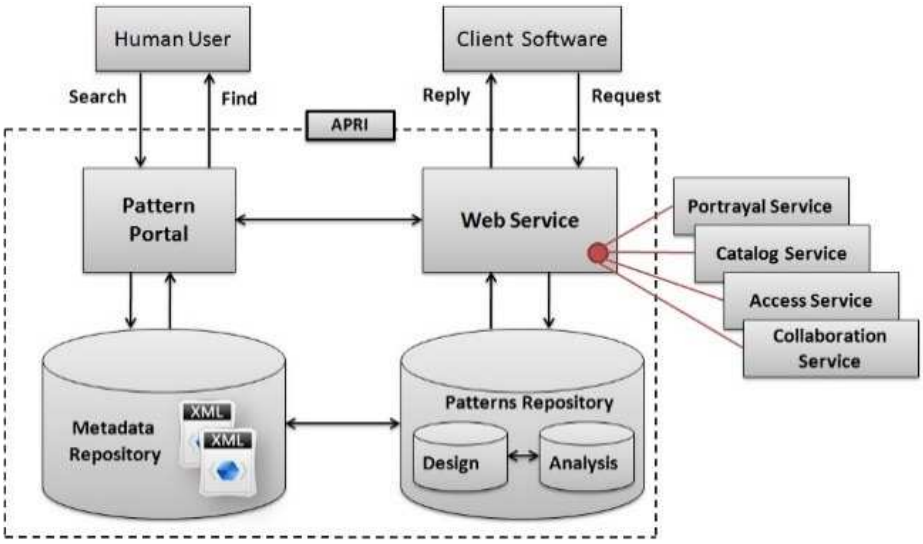

Figure 4. Extension of the Analysis Patterns Reuse Infrastructure (APRI).

5 FINAL CONSIDERATIONS

The DC2DP profile was specified aiming at enabling a more detailed specification of design patterns, since it has been developed specifically to this domain. Through that, this profile must be able to provide users with more complete information, when such information is available, thus helping them to choose the pattern that best aids in carrying out a project and consequent quicker and more effective design production.

With the APRI architecture extension proposed in this paper, the service-oriented infrastructure for cataloging and reusing analysis patterns may be extended and adapted to catalog and reuse design patterns, thus enriching the APRI architecture. This proposal will allow design patterns to be easily found, studied, and reused, also by using web services.

As a future work, an experiment is intended to assess the APRI structure in order to check whether providing these patterns in a repository will in fact help users in their designs. Moreover, an APRI prototype will be created so as to technologically enable the practical application of DC2AP and DC2DP in documenting patterns.

ACKNOWLEDGEMENTS

This project was partially financed by the agencies CAPES, CNPq, FAPEMIG and by the company Funarbe.

<table>
<tr><td colspan="3">1.Identifier: http://purl.org/apri/metadata/Singleton-v1</td></tr>
<tr><td colspan="3">2. Title: Singleton</td></tr>
<tr><td colspan="3">3. Creator: Erich Gamma
Ralph Johnson
Richard Halm
John Vlissides</td></tr>
<tr><td colspan="3">4. Subject: Single instance, Global variable</td></tr>
<tr><td rowspan="4">5.Description</td><td colspan="2">5.1 Problem: Ensure a class only has one instance, and provide a global point of access to it.</td></tr>
<tr><td>5.2 Motivation:
1. It’s important for some classes to have exactly one instance.
2. A global variable makes an object accessible, but it doesn’t keep you from instantiating multiple objects.</td><td>5.2.1Example: 1. Use the Singleton pattern when there must be exactly one instance of a class, and it must be accessible to clients from a well-known access point.
2. When the sole instance should be extensible by subclassing, and clients should be able to use an extended instance without modifying their code.</td></tr>
<tr><td>3. A better solution is to make the class itself responsible for keeping track of its sole instance. The class can ensure that no other instance can be created, and it can provide a way to access the instance.</td><td>5.2.2 Known Uses: 1. The InterViews user interface toolkit [LCI+92] uses the Singleton pattern to access the unique instance of its Session and WidgetKit classes, among others.</td></tr>
<tr><td colspan="2"></td></tr>
<tr><td>6. Date:</td><td colspan="2">6.1 Created: 1995</td></tr>
<tr><td colspan="3">7. Type: Design Pattern</td></tr>
<tr><td colspan="3">8. Language: English – EN</td></tr>
<tr><td rowspan="3">9. History</td><td colspan="2">9.1 Event Date: 1995</td></tr>
<tr><td colspan="2">9.2 Author: Erich Gamma, Ralph Johnson, Richard Halm, John Vlissides</td></tr>
<tr><td colspan="2">9.3 Reason: Creation of this design pattern.</td></tr>
<tr><td rowspan="2">10.Structure Modelling</td><td>10.1Class Diagram:
http://purl.org/apri/patterns/Singleton_v1</td><td>10.1.1 Class Description: Singleton: defines an Instance operation that lets clients access its unique instance. Instance is a class operation (that is, a class method in Smalltalk and a static member function in C++). May be responsible for creating its own unique instance.</td></tr>
<tr><td></td><td>10.1.2 Relation Description: Clients access a Singleton instance solely through Singleton's Instance operation.</td></tr>
<tr><td rowspan="2">11. Sample Code</td><td colspan="2">11.1 Programming Language: C++</td></tr>
<tr><td>11.2 Code:
class MazeFactory {
public:
static MazeFactory* Instance();
// existing interface goes here
protected:
MazeFactory();
private:
static MazeFactory* _instance;
};</td><td>11.3 Code Description: MazeFactory defines an interface for building different parts of a maze. What's relevant here is that the Maze application needs only one instance of a maze factory, and that instance should be available to code that builds any part of the maze. This is where the Singleton pattern comes in. By making the MazeFactory a singleton, we make the maze object globally accessible without resorting to global variables. We make it a Singleton class in C++ by adding a static Instance operation and a static _instance member to hold the one and only
instance. We must also protect the constructor to prevent accidental instantiation, which might lead to more than one instance.</td></tr>
<tr><td rowspan="2">12. Consequences</td><td colspan="2">12.1 Positive: 1. Controlled access to sole instance. Because the Singleton class encapsulates its sole instance, it can have strict control over how and when clients access it.</td></tr>
<tr><td colspan="2">12.2 Negative: Cannot inhibit access your class. Any part of the code to call the method Instance (), because it is static, and have access to the data class.</td></tr>
</table>

Figure 5. Specification of the Singleton design pattern

2.2 Artigo II: APRImora: Uma Arquitetura Baseada em Web Semântica para Recuperação de Informações em Repositórios de Padrões

Angélica Ap. A. Ribeiro, Jugurta Lisboa Filho, Lucas F. da M. Vegi,
Alcione de P. Oliveira, Emílio José de S. Fonseca

RESUMO

Este artigo apresenta a arquitetura APRImora, uma extensão baseada em Web Semântica da Infraestrutura de Reuso de Padrões de Análise (APRI). Além dos padrões de análise, a APRImora agora inclui também suporte ao reuso de padrões de projeto, documentados por meio do perfil Dublin Core para Padrões de Projeto (DC2DP). A arquitetura APRImora tem como objetivo auxiliar na recuperação de padrões de forma mais eficiente, favorecendo assim a disseminação dos padrões e consequentemente a sua reutilização. O artigo descreve ainda o novo módulo para edição de metadados DC2DP.

Palavras-chave: Reuso, Web Semântica, Padrão de Análise, Padrão de Projeto, Artefatos Computacionais

ABSTRACT

This article presents the APRImora architecture, an extension based in Semantic Web of the Analysis Patterns Reuse Infrastructure (APRI). This article also describes an extension of the APRI, to support design patterns, documented by the Dublin Core profile to Design Patterns (DC2DP). The APRImora architecture aims to help in the retrieving of patterns in an efficient way, thus favoring the dissemination of patterns and subsequent its reuse. The article also describes the new module for editing DC2DP metadata.

Keywords: Reuse, Semantic Web, Analysis Pattern, Design Patterns, Computational Artefacts

1 INTRODUÇÃO

Mecanismos de reuso, como padrões de projeto (GAMMA et al., 1995) e padrões de análise (FOWLER, 1997), são artefatos computacionais fundamentais no processo de desenvolvimento de software. No entanto, o grande desafio que um projetista de software encontra ainda hoje é descobrir a existência de um padrão que resolva determinado problema. As máquinas de busca nem sempre ajudam muito, pois embora a busca seja facilitada pelos aperfeiçoamentos constantes dos motores de busca, uma simples pesquisa pode retornar um número elevado de resultados e na maioria das vezes resultados que não são relevantes para o contexto da pesquisa. Por exemplo, ao pesquisar no Google pelo padrão de projeto *Singleton*, digitando apenas a palavra *Singleton,* sem dar nenhuma informação adicional, serão retornados aproximadamente 6.6 milhões de resultados, sendo grande parte destes resultados irrelevantes e inadequados para resolver o propósito da busca.

Segundo Cunha et al. (2011), os dados da Web são organizados para serem lidos ou compreendidos por humanos e não por agentes de software. Isso ocorre devido à forma da Web organizar as informações disponíveis na Internet por meio de hipertextos providos pela linguagem HTML. Esta forma como as informações estão organizadas torna a interação do usuário com a rede mundial mais amistosa, no entanto, a linguagem HTML é utilizada apenas para listar recursos para o usuário, não apresentando o significado destes recursos e não fazendo associações ou correlações de significados.

Para resolver esse problema surge a Web semântica, a qual propõe transformar a Web de documentos na Web de dados, onde todos os dados contidos na Web irão possuir uma semântica associada e estarão ligados entre si. Segundo Cunha (2011) no cenário da Web, isso representaria a atribuição de significado aos dados, interligando-os com outros conjuntos de dados ou domínios de conhecimento, criando assim uma relação de significância entre os conteúdos publicados na Internet, de modo que esses conteúdos possam ser analisados não somente por humanos, mas por agentes de software, resultando em pesquisas mais eficazes e com resultados mais próximos do que o usuário deseja.

Para que artefatos computacionais possam ser mais facilmente recuperados e reutilizados, Vegi et al. (2012a) propuseram a APRI, uma Infraestrutura de Reuso de Padrões de Análise. O objetivo da APRI é possibilitar que os padrões de análise possam ser documentados, disseminados e aprimorados por meio de uma abordagem baseada em

metadados e Web Services. Os padrões de análise adicionados na APRI são documentados por meio um perfil de aplicação do padrão de metadados Dublin Core e salvos no formato RDF de forma a serem acessíveis pelos motores de busca (VEGI et al., 2013).

Este artigo descreve uma proposta de extensão da arquitetura da APRI para uma arquitetura que tenha como base a Web Semântica, transformando-a em uma Infraestrutura Semântica de Reuso de Padrões, denominada APRImora. O restante do artigo está estruturado como segue. A seção 2 apresenta uma revisão sobre as tecnologias da Web Semântica, além de outras tecnologias importantes relacionadas ao trabalho. A seção 3 relaciona alguns trabalhos correlatos. A arquitetura proposta para a APRImora é apresentada na seção 4. A seção 5 descreve o DC2DP *Metadata Editor*, que é um editor de metadados para documentar padrões de projeto na infraestrutura proposta. As considerações finais e alguns trabalhos futuros são descritos na seção 6.

2 REFERENCIAL TEÓRICO

A subseção 2.1 faz considerações sobre Web Semântica e a sua importância para a pesquisa realizada. Os padrões de análise são armazenados na APRI utilizando o formato RDF, sendo assim é necessário utilizar uma linguagem que seja capaz de realizar consultas em documentos RDF. Alguns aspectos da linguagem de consulta SparQL são descritos na subseção 2.2. A subseção 2.3 discorre sobre o conceito de *Linked Data,* uma vez que os padrões armazenados na APRImora são documentados seguindo esse conceito. O uso de Web Services permite que os padrões possam ser reutilizados tanto por humanos quanto de forma automática por ferramentas CASE, dessa forma Web Service são descritos na seção 2.4. Na subseção 2.5 são feitas considerações sobre metadados, especificamente sobre o padrão de metadados Dublin Core, apresentando a importância desse padrão para a pesquisa realizada, uma vez que a sua simplicidade e versatilidade auxilia na construção de perfis para documentar domínios específicos, como é o caso de padrões de análise e de projeto.

2.1 Web Semântica

A Web tem mudado a forma como as pessoas se comunicam e como os negócios são conduzidos. Essa mudança tem transformado o mundo desenvolvido e o encaminhado para uma economia baseada no conhecimento e, mais amplamente em uma sociedade do conhecimento (ANTONIOU et al., 2008).

Segundo Berners-Lee et al. (2001) a Web se desenvolveu mais rapidamente como um meio de documentos para as pessoas, em vez de se desenvolver para os dados e informações que podem ser processados automaticamente. Como consequência deste fato, ao realizar uma pesquisa, na maioria das vezes é obtido como resultado milhares de ocorrências com pouca ou nenhuma relevância. Isso ocorre devido ao significado do conteúdo existente na Web não ser processável por máquina, além disso a capacidade dos softwares é limitada no que diz respeito à interpretação de frases e extração de informações úteis para os usuários (ANTONIOU et al., 2008).

Diante disso surgiu a Web Semântica que tem por objetivo compensar esse problema através do fornecimento de meios para organizar os dados e permitir que esses possam ser interpretados por computadores. De acordo com Hortêncio-Filho et al. (2009), com os recursos providos na Web semântica, é possível melhorar os resultados de busca e facilitar a automação de tarefas relevantes, evitando assim o desperdício de tempo do usuário. Berners-Lee et al. (2001) ainda acrescentam que a Web Semântica não é uma Web separada, mas uma extensão da atual, onde a informação possui um significado bem definido, permitindo assim que computadores e pessoas trabalhem em cooperação.

Para Berners-Lee et al. (2001), o desafio da Web Semântica é fornecer uma linguagem que expresse tanto os dados quanto as regras para raciocínio sobre estes dados, e que permita que as regras de qualquer sistema de conhecimento existente possam ser exportadas para a Web. A Figura 1 ilustra as camadas que compõem a Web Semântica. Detalhes sobre cada camada podem ser encontrados em (Ramalho et al., 2007).

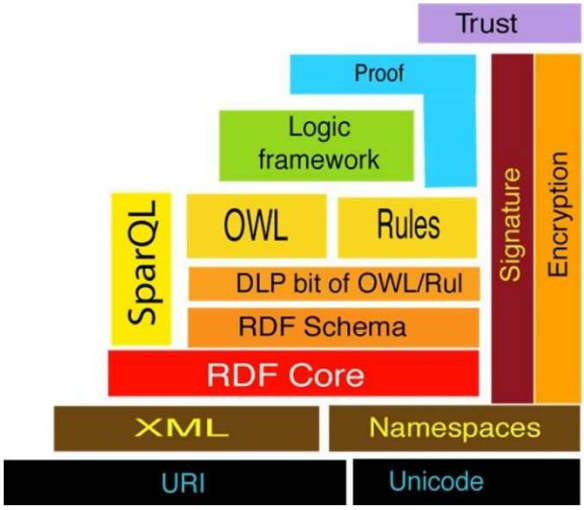

Figura 1: Arquitetura da Web Semântica
Fonte: Berners-Lee (2005)

2.2 SparQL

O formato RDF é uma forma flexível e extensível a ser utilizado para representar informações sobre recursos da Web, por exemplo, informações pessoais, metadados sobre artefatos digitais como música e imagens, entre outros. Além de fornecer meios de integração entre diferentes fontes de informação. Uma linguagem de consulta padronizada para busca em dados RDF oferece aos desenvolvedores e usuários finais uma maneira de escrever e de consumir os resultados de consultas em toda esta vasta gama de informações (BECKETT et al., 2013).

Assim como os bancos de dados precisam de linguagens específicas para realizar consultas em sua base dados, a Web representada usando RDF como um dos formatos de dados, precisa de sua própria linguagem de consulta (W3C, 2014). *SPARQL Protocol and RDF Query Language* (SparQL) é uma linguagem e um protocolo de acesso a dados desenvolvida e recomendada pela W3C para a realização de consultas em arquivos RDF.

A Figura 2 ilustra uma consulta SparQL extraída de Harris et al. (2013), para auxiliar na compreensão dessa linguagem e o uso da sua sintaxe. A Figura 2(a) exibe uma tripla correspondendo ao dado. A Figura 2(b) ilustra a consulta, onde a cláusula *SELECT* identifica as variáveis que irão aparecer no resultado da pesquisa e a cláusula *WHERE* fornece o padrão básico a ser combinado com o gráfico de dados, esse padrão básico nesse exemplo consiste de uma variável única (?title) na posição do objeto. A Figura 2(c) mostra o resultado da consulta realizada.

```
<http://example.org/book/book1>
<http://purl.org/dc/elements/1.1/title> "SPARQL Tutorial" .
                                (a)

SELECT ?title
WHERE
{
<http://example.org/book/book1><http://purl.org/dc/elements/1.1/t
itle> ?title .
}
                                (b)

"SPARQL Tutorial"
                                (c)
```

Figura 2: (a) Tripla RDF, (b) Consulta SparQL, (c) Resultado da consulta
Fonte: Harris et al. (2013)

2.3 Linked Data

O objetivo da Web Semântica é fornecer meios para organizar os dados e permitir que esses dados possam ser interpretados por computadores, tornando o trabalho mais útil, além de desenvolver sistemas que possam apoiar interações confiáveis na rede (W3C, 2014). Além disso, a Web Semântica é vista como uma camada da Web onde é possível publicar, obter e utilizar dados que podem ser processados direta ou indiretamente por máquinas (BERNERS-LEE, 2000).

Mas a sua importância vai além de apenas disponibilizar os dados na Web, a Web Semântica realiza ligações entre esses dados de forma a permitir que uma pessoa ou uma máquina possa explorá-los. Através da interligação dos dados é possível encontrar mais informações relacionadas a um determinado dado pesquisado (BERNERS-LEE, 2006). Essa ligação fornece meios para que Web convencional possa ser transformada em uma Web de dados processáveis por máquinas, e a essa ligação dá-se o nome de *Linked Data*.

Linked Data refere-se ao conjunto de melhores práticas para a publicação e conexão de dados na Web. Essas práticas levaram à criação de um espaço global de dados que contém bilhões de informações, a chamada Web de Dados (BIZER et al., 2009). Segundo Berners-Lee (2006) assim como a Web de hipertexto, a Web de dados é construída a partir de documentos na Web. Contudo, a diferença entre elas é que na Web de hipertexto os links relacionados estão ancorados em documentos escritos em HTML. Já na Web de dados as ligações entre coisas arbitrárias são realizadas por meio de descrição em RDF utilizando *Uniforme Resource Identifiers* (URIs). As URIs identificam qualquer tipo de objeto ou conceito.

A URI e o formato RDF são tecnologias que auxiliam o *Linked Data* a dar suporte a Web Semântica. A URI providencia uma maneira simples e extensível para identificar através de uma sequência de caracteres um recurso abstrato ou físico de forma única (BERNERS-LEE, 2005). O RDF é um framework para representar informações na Web, sua estrutura é composta de um conjunto de triplas denominadas de grafo RDF. Esse grafo RDF é formado pelos conceitos de sujeito, predicado e objeto, onde sujeito e objeto são identificadores URI que representam os respectivos dados relacionados e o predicado especifica a semântica desse relacionamento, ou seja, o tipo de relacionamento existente entre os dados (KLYNE et al., 2014).

2.4 Web Service

Web Service é uma solução utilizada na integração de sistemas e na comunicação entre aplicações diferentes. Uma das suas vantagens é que a comunicação entre os serviços é padronizada possibilitando assim a independência de plataforma e de linguagem de programação, permitindo que um sistema feito em Java e rodando em um servidor Linux possa acessar, com transparência, um serviço implementado em .Net rodando em um servidor Microsoft (PAMPLONA, 2006).

Segundo Doyle et al. (2001) Web Services são aplicações Web independentes, auto descritivas e modulares que podem ser publicadas, localizadas e chamadas por toda a Web. Utiliza de XML para codificar e decodificar os dados e *Simple Object Acess Protocol* (SOAP) para transportá-los (W3SCHOOLS, 2014).

Web Service Semântico (SWS), como o próprio nome indica, é a interseção de duas tendências importantes na evolução da Web, sendo elas, os serviços Web e a Web Semântica (MARTIN et al., 2007). Segundo Martin et al. (2007) o tema principal da SWS é o uso de descrições mais ricas e declarativas de elementos da dinâmica da computação distribuída. Estas descrições permitem uma completa automação, uma prestação de serviço mais flexível e a utilização e construção de ferramentas mais poderosas e melhores metodologias para trabalhar com serviços.

A representação mais rica da estrutura permite uma especificação mais abrangente de muitos aspectos dos serviços, com isso SWS podem fornecer uma base sólida para uma grande quantidade de atividades durante o ciclo de vida do serviço Web (MARTIN et al., 2007).

Martin et al. (2007) destaca quatro pontos que uma descrição mais rica dos serviços pode suportar, são elas: uma maior automação na seleção de serviços e invocação; tradução automática de conteúdo de mensagens entre serviços heterogêneos; abordagens automatizadas ou semiautomatizadas para composição de serviços; e abordagens mais abrangentes para o serviço de monitoramento e recuperação de falha.

2.5 Padrão de metadados Dublin Core

O conjunto de elementos do padrão de metadados Dublin Core, recebeu o nome "Dublin" uma vez que sua criação ocorreu em 1995 em um workshop realizado na cidade de

Dublin no estado de Ohio, USA. “Core” porque seus elementos são básicos e genéricos, possibilitando a sua utilização para descrever uma ampla gama de recursos (NISO U. S., 2007). Devido a sua versatilidade, com o padrão de metadados Dublin Core (DC) é possível documentar desde obra de artes, a livros em uma biblioteca.

O padrão Dublin Core é composto por um conjunto de quinze elementos. São eles (DCMI, 1999): *title* (o nome dado ao recurso), *creator* (pessoa ou organização responsável pelo conteúdo), *subject* (assunto coberto pelo documento), *description* (descrição do conteúdo), *publisher* (entidade responsável por tornar o conteúdo disponível), *contributor* (entidade responsável por realizar contribuições ao conteúdo), *date* (data associada a um evento ocorrido no ciclo de vida do conteúdo), *type* (a natureza ou gênero do conteúdo), *format* (formato no qual o recurso é apresentado), *identifier* (um identificador único para o conteúdo), *source* (fonte de origem do conteúdo), *language* (a linguagem que o conteúdo foi escrito), *relation* (relacionamento do recurso com outros recursos), *coverage* (característica espacial e temporal do recurso), *rights* (informação sobre direitos sobre o recurso).

Embora seja um padrão de metadados simples e que não ofereça, isoladamente, recursos suficientes para descrever dados de domínios complexos (NISO U. S., 2001), a versatilidade do Dublin Core em documentar artefatos de diversos tipos se faz por meio de extensões, conhecidas como perfis de aplicação. Desde 2000 a comunidade Dublin Core vem propondo perfis de aplicação, onde a ideia principal é a adequação dos elementos do Dublin Core para gerar novos conjuntos de metadados para atender aos requisitos específicos de aplicações de diferentes domínios (DCMI, 2014).

No contexto da APRI, foram definidos dois perfis de metadados Dublin Core, o DC2AP (VEGI et al., 2012a) para documentação de Padrões de Análise e o DC2DP (RIBEIRO et al., 2014) para documentação de Padrões de Projeto.

Segundo Vegi et al. (2012a), o DC2AP foi desenvolvido a partir do mapeamento dos elementos contidos no padrão de metadados Dublin Core com os elementos do *template* proposto no trabalho de Pantoquilho et al. (2003) para documentar padrões de análise. O DC2AP é um perfil de aplicação processável por máquina, o que possibilita que os padrões de análise sejam descritos e publicados como *Linked Data* por meio de arquivos no formato RDF. O DC2AP tem como principal objetivo melhorar a recuperação e reutilização dos padrões de análise através de uma descrição que permita um tratamento mais preciso

realizado por um computador, oferecendo informações detalhadas sobre os padrões de análise que antes não eram recuperados por ferramentas de busca.

O perfil de Aplicação Dublin Core para documentar Padrões de Projeto (DC2DP), proposto por Ribeiro et al. (2014), é um perfil de aplicação do padrão de metadados Dublin Core criado especificamente para descrever padrões de projeto de maneira compatível com a proposta da APRI. Os elementos do DC2DP foram definidos a partir da combinação de três conjuntos de metadados, sendo eles os elementos do padrão de metadados Dublin Core, os elementos do *template* de Gamma et al. (1995) e os elementos do DC2AP. O Dublin Core é a base do DC2DP, portanto todos os elementos pertencentes a esse padrão de metadados integram o perfil DC2DP. O *template* de Gamma et al. (1995) contribuiu com os elementos específicos para documentar padrões de projeto. Já os elementos de controle de versão foram reaproveitados do perfil DC2AP.

Como o DC2AP, o DC2DP também é processável por máquina, possibilitando que os padrões de projeto sejam descritos e publicados como *Linked Data* através de arquivos no formato RDF, permitindo assim que a recuperação e a reutilização dos padrões sejam realizadas de forma mais precisa por computador. Uma descrição técnica detalhada destes dois perfis de aplicação, contendo a descrição semântica de cada um dos seus elementos, bem como detalhes das regras de aplicação dos mesmos, pode ser encontrada em http://www.dpi.ufv.br/projetos/apri/.

3 TRABALHOS RELACIONADOS

Monteiro (2013) propõe uma arquitetura baseada em Web Semântica para repositórios digitais educacionais na área de saúde, que disponibilizam objetos de aprendizado com aspectos educacionais descritos em formulários de metadados. Objetos de aprendizado (OA) são recursos estruturados utilizados para disponibilizar conteúdo para ensino-aprendizagem e são adequados para o desenvolvimento de conteúdos para Educação à Distância. A disponibilização de OA na Web contribui com a redução de gastos com sua produção. Para permitir a disponibilização de OA são constituídas coleções no *Learning Management System* (LMS) para que os OA sejam acessados no próprio ambiente dos cursos. Também podem ser desenvolvidos Repositórios Digitais Educacionais (RDE) onde um número cada vez maior de objetos é compartilhado. Por esse motivo, o RDE deve possuir uma arquitetura adequada e dispor de informações descritivas suficientes à seleção de OA, especialmente no que diz respeito aos aspectos educacionais.

Assim, Monteiro (2013) propõe a adição de tecnologias na arquitetura de RDE capazes de refinar o resultado da busca em OA, utilizando para isso a Web Semântica como ponto crucial na contribuição para o refinamento de busca. A arquitetura proposta é apresentada na Figura 3.

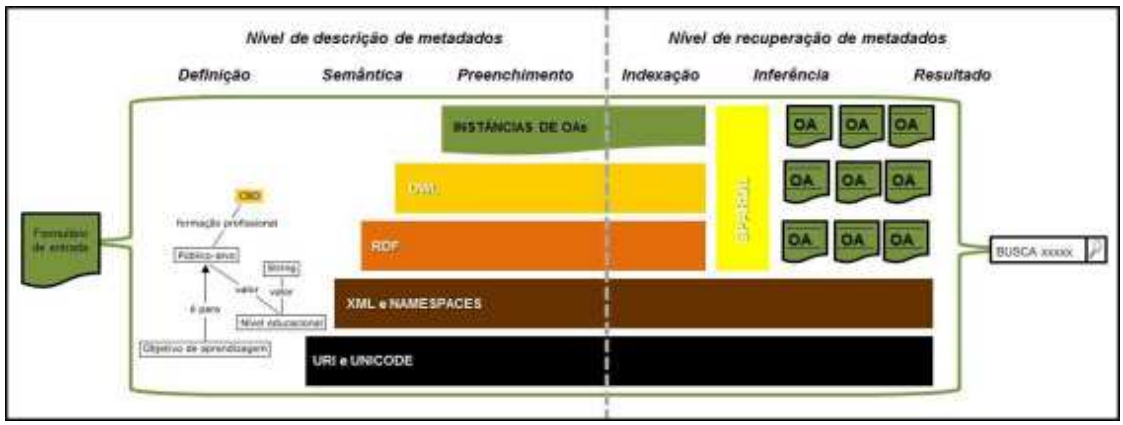

Figura 3: Arquitetura de busca em RDE
Fonte: (Monteiro, 2013)

A arquitetura apresentada na Figura 3 mostra como deve ser realizada a modelagem para a implementação de repositórios digitais educacionais. Esta arquitetura está dividida em dois níveis diferentes, porém complementares: o Nível de Descrição e o Nível de Recuperação. Cada nível contém especificações da tecnologia Web Semântica capaz de auxiliar na recuperação das informações em repositórios educacionais.

O Nível de Descrição indica como a informação que descreve o objetivo de aprendizagem que é uma informação descritiva, registrada nos metadados, é representada com as tecnologias da Web Semântica. Esse nível é composto das seguintes camadas:

- *Definição de metadados* - é a sintaxe dos conceitos que especificam o objetivo de aprendizagem e são representados por metadados no RDE. Para a definição de metadados são utilizados XML e XML Schema;
- *Semântica de metadados* - é a formalização da relação semântica entre os metadados que possibilitam inferências na recuperação da informação. Para a semântica de metadados são utilizados o RDF e o RDF Schema;
- *Preenchimento de metadados* - Delimita valores de preenchimento dos metadados e armazenamento de instância que descrevem OA. Essa delimitação utiliza OWL para estabelecer disjunções, equivalências e cardinalidade.

O Nível de Recuperação indica como a informação é recuperada para apresentar resultados de buscas refinados, conforme o objetivo da pesquisa. Esse nível é composto por:

- *Indexação* - os valores de preenchimento são indexados para possibilitar a busca;
- *Inferência* - é a identificação e interpretação da semântica da informação que descreve o OA relacionando-o com o objetivo de aprendizagem;
- *Resultado* - é a verificação dos termos da estratégia de busca dos usuários nos valores de preenchimento de metadados indexados e a realização de inferências para apresentar resultados refinados. Para a recuperação semântica são considerados inferências RDF e OWL, sendo necessário a utilização da linguagem de consulta SparQL.

Outro trabalho diretamente relacionado à proposta apresentada nesse artigo é a Infraestrutura de Reuso de Padrões de Análise (APRI), proposta por Vegi et al. (2012c). A APRI é uma estrutura inspirada nas Infraestruturas de Dados Espaciais (IDE), que por sua vez são definidas como coleções de tecnologias, políticas e arranjos institucionais que facilitam a disponibilidade e o acesso aos dados espaciais (NEBERT, 2004). Muitas IDEs empregam o conceito de arquitetura orientada a serviço (SOA), permitindo assim, que possam ser desenvolvidos ambientes compartilhados, distribuídos e interoperáveis com base em Serviços Web (DAVIS JÚNIOR et al., 2005).

De maneira análoga à IDE a APRI utiliza metadados para descrever os padrões de análise a serem armazenados em seu repositório. Com isso o objetivo da APRI é fornecer mecanismos para que os padrões de análise sejam disponibilizados e encontrados pelos projetistas, auxiliando assim na difusão desses padrões e consequentemente seu reuso. A Figura 4 apresenta a arquitetura da APRI. Uma descrição detalhada da APRI pode ser encontrada em (VEGI et al., 2012c).

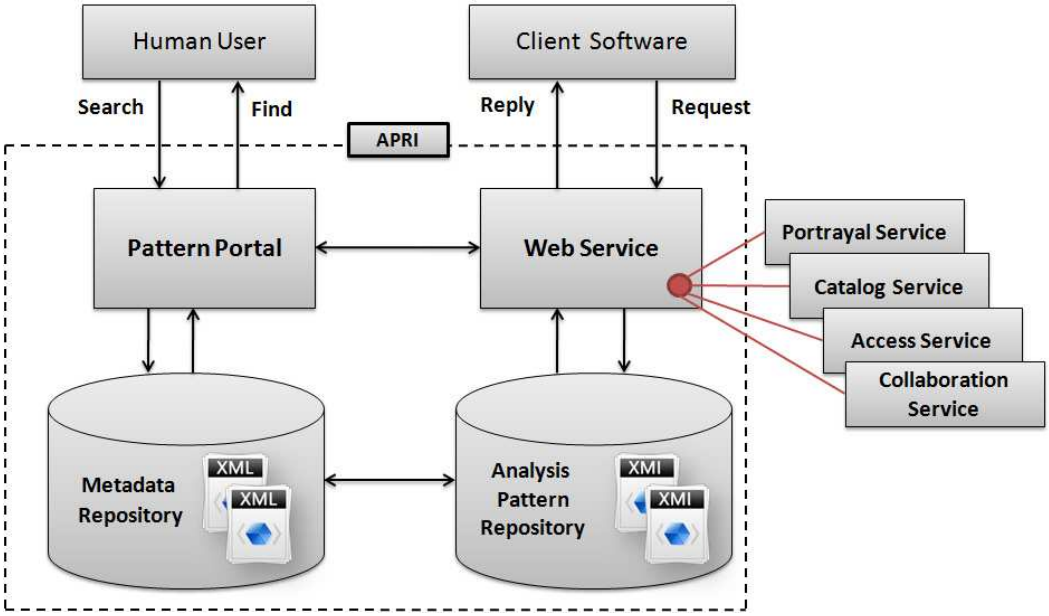

Figura 4: Arquitetura da APRI
Fonte: (VEGI et al., 2012c)

4 APRIMORA: UMA ARQUITETURA APOIADA EM WEB SEMÂNTICA PARA REUSO DE PADRÕES

Inicialmente, a APRI foi desenvolvida com foco apenas no reuso de padrões de análise (VEGI et al., 2012b). Posteriormente, Ribeiro et al. (2014) propuseram a adição de padrões de projeto nos repositórios da APRI, ampliando assim seu escopo de aplicação, permitindo a reutilização de padrões, não só durante a etapa de análise do desenvolvimento de software, mas também na etapa de projeto.

Os padrões armazenados no repositório da APRI podem ser recuperados a partir de um navegador Web e também por meio de serviços que proporcionam a descoberta, catalogação e reuso de padrões a partir do *Pattern Portal* (Figura 4). No entanto, por se utilizar máquinas de busca, o sistema pode retornar uma quantidade de resultados irrelevantes para o contexto da pesquisa e inadequados para resolver o propósito da busca.

Os padrões documentados na APRI estão descritos como *Linked Data*, o que auxilia na sua recuperação. Porém, apenas isso não é suficiente para a recuperação de forma aprimorada dos padrões contidos nos repositórios da APRI, uma vez que os metadados possuem uma semântica formal limitada. Segundo Arenas et al. (2013), o método mais comum para procurar um conjunto de dados consiste em buscar os elementos por palavras-chaves que tenham correspondência nos metadados.

O fato das buscas baseadas em metadados retornarem uma grande quantidade de resultados fora do contexto da pesquisa, devido a utilização de máquinas de busca, torna necessária a definição da semântica dos elementos. A semântica é que dá o significado aos elementos e é um ponto essencial para auxiliar na recuperação da informação nos repositórios de forma precisa. Portanto, este trabalho estende a APRI para uma arquitetura que tenha como base a Web Semântica, capaz de auxiliar o usuário na recuperação aprimorada de padrões contidos em seus repositórios, dando origem à APRImora, uma Infraestrutura Semântica de Reuso de Padrões (Figura 5).

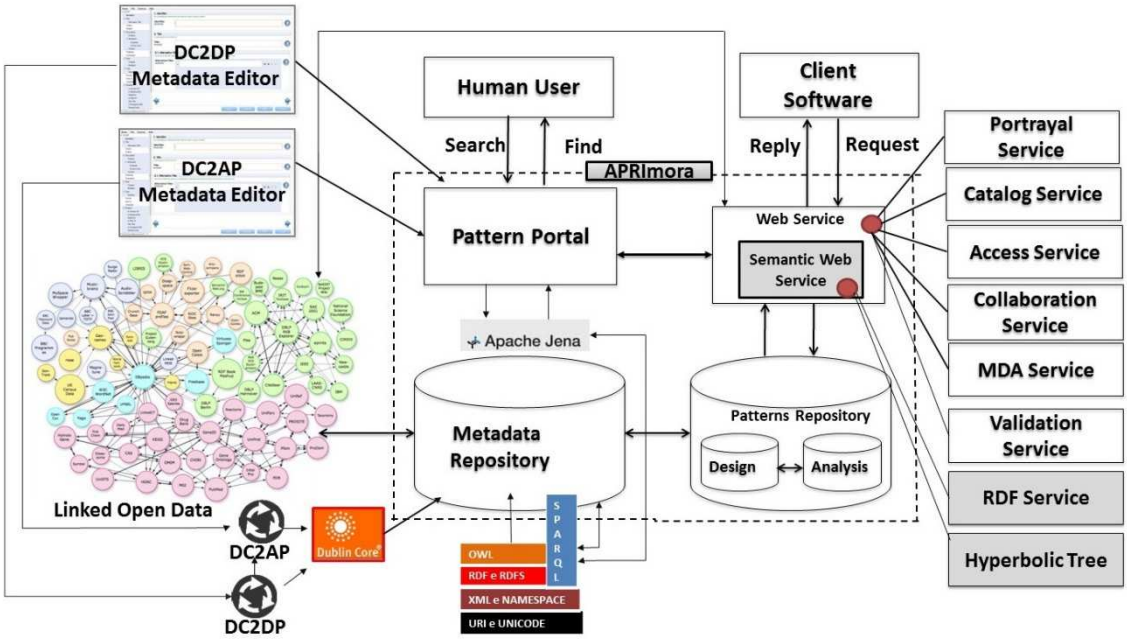

Figura 5: Arquitetura da APRImora

A arquitetura da APRImora tem como núcleo central os mesmos componentes da APRI, o que muda é a adição de semântica aos elementos e a forma como estes são recuperados e adicionados na estrutura. As subseções seguintes descrevem os principais componentes dessa arquitetura.

4.1 Pattern Portal

O *Pattern Portal* é um portal composto por um conjunto de Web sites focados na obtenção de padrões, além de fornecer ferramentas e serviços que proporcionam a descoberta, catalogação e reuso dos mesmos. Algumas ferramentas disponíveis no *Pattern Portal* são o editor do perfil de metadados para padrões de análise (*DC2AP Metadata Editor*) (PEIXOTO

et al., 2013) e o editor do perfil de metadados para padrões de projeto (*DC2DP Metadata Editor*).

Cada elemento que compõe o conjunto de metadados destas ferramentas possui regras relativas à sua obrigatoriedade, ocorrência e tipo de valor, que devem ser validadas para que o padrão documentado seja consistente. Após documentar o padrão de análise ou de projeto, as regras de obrigatoriedade referentes a esses metadados são validados por um serviço Web chamado *Validation Service*. Esse serviço verifica, por exemplo, se todos os campos que possuem como regra de ocorrência *Obrigatório* [M] foram preenchidos. Caso contrário, o *Validation Service* retornará uma informação de erro ao usuário.

As regras de ocorrência, que dizem respeito à multiplicidade dos elementos, têm a sua validação realizada pela própria interface. Essa validação ocorre por meio de botões que adicionam mais campos a elementos que devem ser múltiplos e os omitem em campos que devem ser simples.

Alguns elementos são preenchidos e validados por vocabulários controlados, que são conjuntos de termos padronizados que auxiliam na entrada e saída de dados em um sistema de informação, promovendo assim maior precisão e eficácia na comunicação entre os usuários e o sistema de informação (KOBASHI, 2008). Exemplos de elementos com vocabulário controlado são *type*, *format*, *language* (DC2AP) e *type* e *language* (DC2DP).

Quando o *Validation Service* retorna uma resposta afirmativa, o *RDF Service* pode ser utilizado para gerar um documento RDF do novo padrão. Documentos RDF são armazenados de forma automática no *Metadata Repository*, em uma pasta definida pelo usuário.

Além das ferramentas de edição de metadados, o framework Apache Jena faz parte do *Pattern Portal.* Apache Jena ou simplesmente Jena é um *framework* em Java gratuito e de código aberto para a construção de aplicações Web Semântica e *Linked Data*. Esse *framework* é composto de diferentes APIs interagindo entre si para processar dados RDF (APACHE JENA, 2014).

Na arquitetura APRImora esse framework é utilizado para recuperar informações nos repositórios de dados e metadados, juntamente com a linguagem SparQL, utilizando o módulo chamado ARQ. ARQ é um mecanismo de consulta utilizado por Jena que possui suporte à linguagem SparQL (APACHE JENA, 2014).

4.2 Camadas da Web Semântica

As tecnologias URI, XML e *Namespace*, RDF e RDFS, e OWL, pertencentes ao conjunto de camadas da Web Semântica, são utilizadas diretamente nos documentos contidos no *Metadata Repository.* Outras camadas da Web Semântica (Figura 1), como *Logic Framework, Proof*, *Trust*, *Signature* e *Encryption* não foram adicionadas à estrutura da APRImora uma vez que ainda não possuem recomendações por parte da W3C para a utilização dessas camadas (MONTEIRO, 2013).

A primeira camada da Web Semântica, a camada URI, é utilizada para nomear de forma única os elementos, utilizando o padrão Unicode, que é o padrão universal adotado para esse endereçamento. A utilização de URI garante a unicidade dos elementos, evitando assim a existência de ambiguidade.

O endereçamento URI é utilizado na APRImora com o objetivo de nomear de forma única os padrões a serem documentados. O campo *Identifier* é responsável por receber o valor da URI, indicando a unicidade dos padrões documentados por meio dos metadados DC2AP ou DC2DP. Além disso, a URI é utilizada para relacionar outros padrões ao que está sendo documentado.

Na APRImora o campo *Identifier* recebe um endereço URI informado pelo usuário, gerado utilizando *Persistent Uniform Resource Locators* (PURLs)[1]. PURLs são endereços da Web que agem como identificadores permanentes, ou seja, possuem meios que permitem que os recursos mudem com o tempo sem afetar negativamente os sistemas que deles dependem. Assim, os endereços Web podem migrar de um domínio para outro sem perder a referência do recurso alocado (PURL, 2014). Para a geração de uma URI, o usuário deve possuir conta de administrador no site purl.org, preencher os campos referente ao PURL que deseja criar e aguardar a verificação se a URI que deseja registrar está liberada para uso, ou se já foi utilizada por outro usuário.

A segunda camada é composta pelas tecnologias XML e *Namespace*. Essa camada permite que a sintaxe do conteúdo a ser disponibilizado na Web seja definida e sua estrutura especificada (MONTEIRO, 2013). Como citado anteriormente, a APRImora utiliza vocabulários e sintaxe controlada em alguns elementos dos perfis de metadados DC2AP e DC2DP. A utilização dessa camada na APRImora permite que os metadados sejam definidos,

[1] http://purl.org/docs/index.html

caracterizando assim a sintaxe dos elementos que compreendem os perfis DC2AP e o DC2DP.

A linguagem XML permite a definição da sintaxe do conteúdo disponibilizado na Web, ou seja, segundo Monteiro (2013), o XML é capaz de definir um texto na Web, seu conteúdo, sua forma de apresentação e os dados referentes a esse conteúdo, como por exemplo o "autor", porém não é possível inferir se o "autor" é uma pessoa ou uma instituição, e nem relacionar textos pertencentes ao mesmo "autor".

Para resolver essa limitação é proposta a utilização do modelo de dados RDF, apresentado na terceira camada da arquitetura da Web Semântica. O modelo de dados RDF é capaz de prover a descrição de sentenças sobre certos domínios de interesse (HORTÊNCIO-FILHO, 2009). Em um documento RDF não são tratados tipos de dados, mas sim suas propriedades ou unidades de informação, dando dessa forma um passo à frente na sintaxe e estruturação dos documentos disponibilizados na Web, porém esse avanço não é o suficiente uma vez que ainda não permite a conceituação e desambiguação dos termos (MONTEIRO, 2013). Na APRImora a camada RDF é utilizada com o objetivo de formalizar a semântica dos metadados.

A quarta camada, também conhecida como camada de ontologia, é utilizada para prover uma maior expressividade para descrever um domínio de interesse. A implementação da camada de ontologia é feita com o uso da linguagem OWL. A OWL foi projetada para ser usada por aplicações computacionais que necessitam processar conteúdo da informação em vez de apenas apresentá-los aos seres humanos (MONTEIRO, 2013). Essa linguagem amplia a capacidade de inferências auxiliadas por XML, RDF e RDFS, fornecendo assim um vocabulário adicional juntamente com uma semântica formal. Na APRImora a camada OWL é utilizada para auxiliar no preenchimento dos metadados.

SparQL é a linguagem de consulta utilizada para recuperar informações a partir de um documento RDF. Segundo Monteiro (2013), SparQL fornece protocolos para consultar e manipular grafos RDF, possibilitando dessa forma diversas inferências. Essa linguagem é utilizada perpendicular as camadas RDF e OWL. Na APRImora a recuperação de informação nos repositórios utilizando a linguagem SparQL é realizada no módulo ARQ pertencente ao framework Apache Jena.

4.3 Linked Open Data

Os padrões documentados e armazenados no *Metadata Repository* são descritos utilizando o conceito de *Linked Data,* gerando entre os padrões existentes nesse repositório uma nuvem de *Linked Data,* como ilustra a Figura 6, onde os nodos em tom de cinza claro indicam padrões de projeto e os nodos em tom de cinza escuro os padrões de análise. Essa nuvem de *Linked Data* representa as ligações realizadas através de URI entre os próprios padrões de projeto e entre os padrões de projeto e de análise.

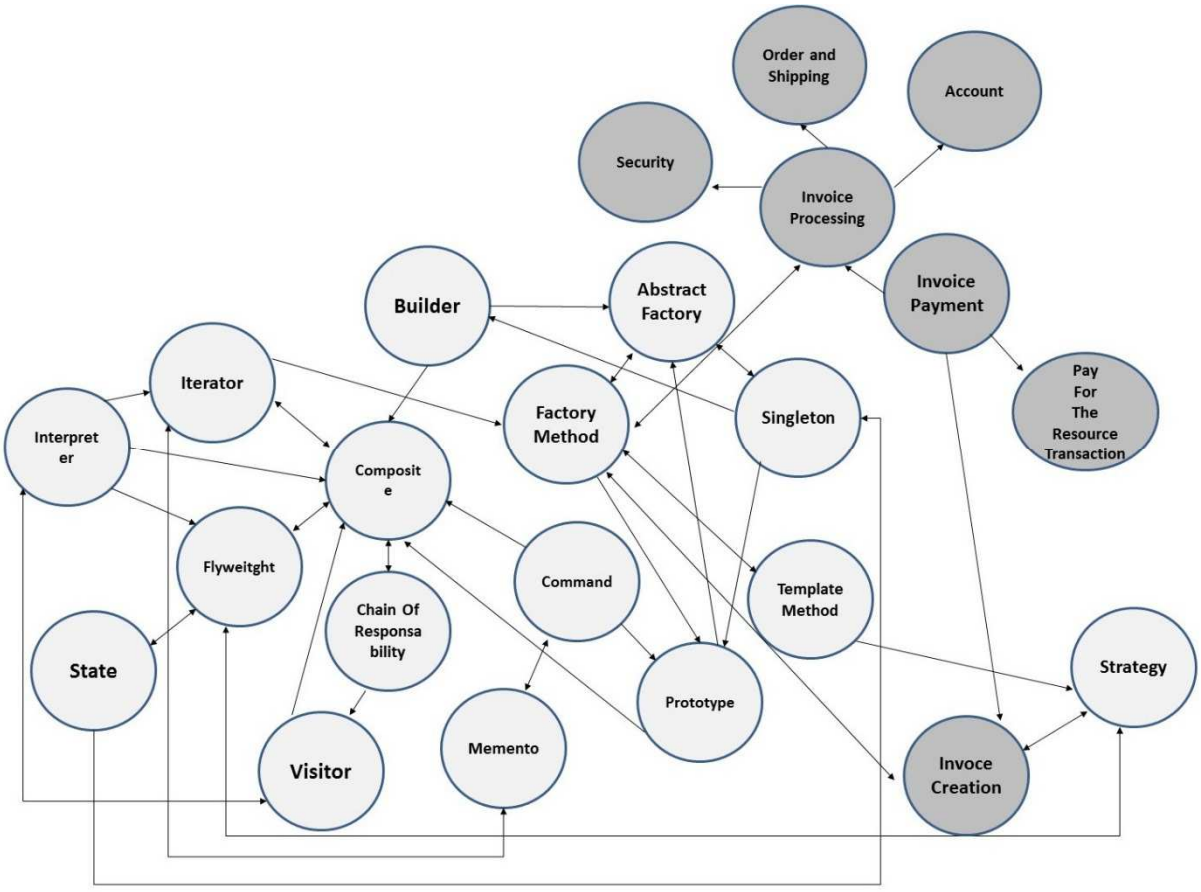

Figura 6: Nuvem de *Linked Data* de parte do repositório de Metadados da APRImora

As ligações apresentadas na Figura 6 indicam que um padrão está relacionado a outros. Segundo Gamma et al. (1995) alguns padrões podem ser utilizados em conjunto, por exemplo, o *Composite* é constantemente utilizado em conjunto com o *Iterator* ou *Visitor*. Alguns padrões podem ser utilizados como alternativas, como é o caso do *Prototype* utilizado de forma alternativa ao *Abstract Factory.* Além disso, alguns padrões podem resultar em projetos semelhantes, embora suas intenções sejam diferentes, como os diagramas de estrutura do *Composite* e *Decorator* são semelhantes.

A principal vantagem das relações existentes entre os padrões é permitir que o projetista tenha múltiplas maneiras de pensar a respeito dos padrões, aprofundando assim sua percepção sobre o que cada padrão faz, como se comparam e em qual situação ele pode ser aplicado (GAMMA et al., 1995). Essas múltiplas maneiras permitem que o projetista possa escolher a melhor forma de utilizar o padrão e aqueles que o complementam, tornando a execução de projetos melhores à medida que reutiliza os padrões.

O *Linked Open Data*, de acordo com Zaidan (2012) é um esforço mundial para publicação de dados, tornando-os abertos e disponíveis para serem utilizados por todo mundo. Para que esses dados sejam publicados e conectados é necessário a utilização de *Linked Data*. Dessa forma, o *Linked Open Data* foi adicionado à arquitetura da APRImora, por meio de uma ligação com o *Metadata Repositor*y, para indicar que a nuvem de *Linked Data* gerada pelos padrões documentados e armazenados nesse repositório passam a integrar o *Linked Open Data*, o que torna os padrões abertos e disponíveis para serem utilizados.

4.4 Pattern Repository

O *Pattern Repository* é um repositório que contém descrições de diagramas expressos em XMI (*XML Metadata Interchange*), JPG e PNG, que representam as soluções propostas pelos padrões, permitindo dessa forma a utilização dos mesmos por serviços de visualização e colaboração. Com a proposta da APRImora, o *Pattern Repository* passa a ser composto por dois repositórios, *Design* e *Analysis,* que se relacionam por meio de *Linked Data.* Dessa forma, um padrão de projeto pode ser referenciado por um padrão de análise e vice-versa.

A correspondência existente entre os padrões de análise e de projeto tem como vantagem o compartilhamento de experiência da utilização desses padrões, ou seja, o analista pode relatar quais padrões de projeto podem ser utilizados para implementar determinado padrão de análise. O mesmo ocorre em relação aos padrões de projetos, onde o projetista pode indicar quais padrões de análises aquele padrão de projeto pode implementar.

4.5 Web Service e Semantic Web Service

A pesquisa na APRImora pode ser realizada tanto por *Human User* por meio de ferramentas de busca, quanto por *Client Software* por meio de *Web Service* ou *Semantic Web Service*.

A busca realizada por *Human User* segue o seguinte fluxo, como mostrado na Figura 7. O *Client Software* realiza a busca por padrões utilizando *Web Service* e o *Semantic Web Service*.

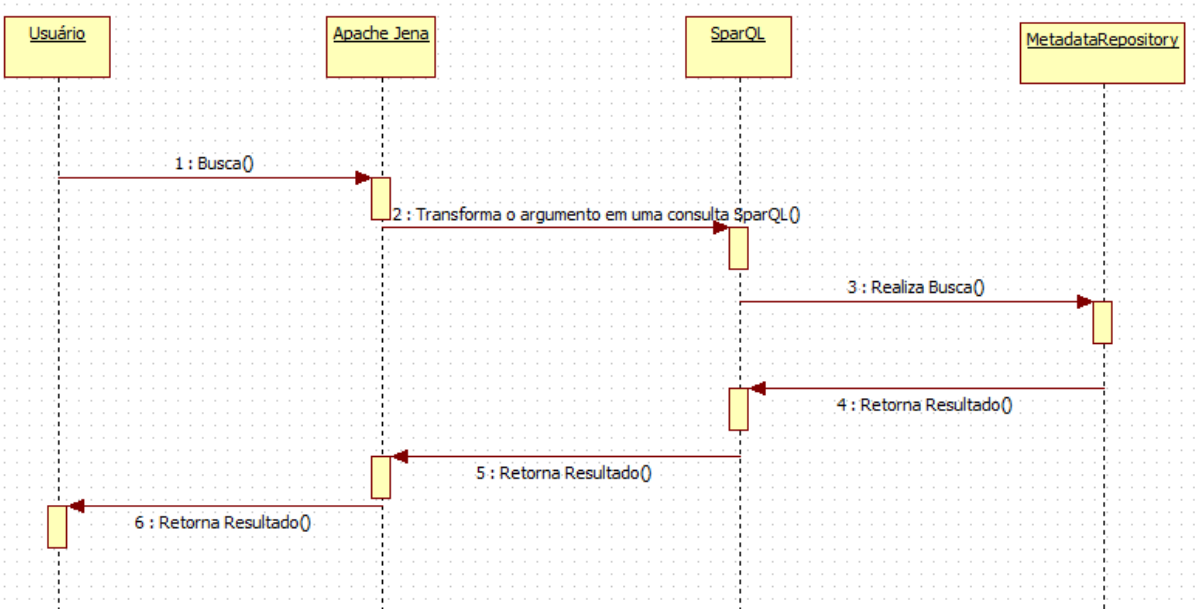

Figura 7: Fluxo de busca realizada por um Human User

Como pode ser observado na Figura 5 a ligação existente entre *Web Service*/*Semantic Web Service e Pattern Portal* é uma ligação de via dupla, uma vez que os serviços podem utilizar ferramentas para realizar as buscas. E o *Pattern Portal* utiliza esses serviços juntamente com suas ferramentas. Por exemplo, os editores de metadados fazem uso de *Web Services* para validar (*Validation Service*) os campos dos metadados, visualizar os diagramas (*Portrayal Service*) e realizar download dos arquivos RDF (*Access Service*) que contêm as descrições dos padrões. Já o *Semantic Web Service* é utilizado pelo *Pattern Portal* para gerar um documento RDF (*RDF Service*) e na geração de árvore hiperbólica (*HyperbolicTtree*).

Com o resultado da busca retornado, o usuário pode utilizar um Seman*tic Web Service* para gerar uma árvore hiperbólica. Uma árvore hiperbólica é um grafo de fácil entendimento sobre como os padrões estão relacionados. No contexto da APRImora, o nó central dessa árvore sempre será o padrão retornado de acordo com os critérios de busca, os demais nós ligados ao nó central são os padrões relacionados a ele com seus respectivos nós. Isso é possível graças ao uso de *Linked Data*.

A Figura 8 ilustra uma árvore hiperbólica correspondente as ligações de possíveis padrões de projeto existentes no repositório de padrões, tendo como nó raiz o padrão

Prototype. A vantagem de se utilizar uma árvore hiperbólica é o fato desta possibilitar ao usuário navegar visualmente entre as ligações existentes entre os padrões.

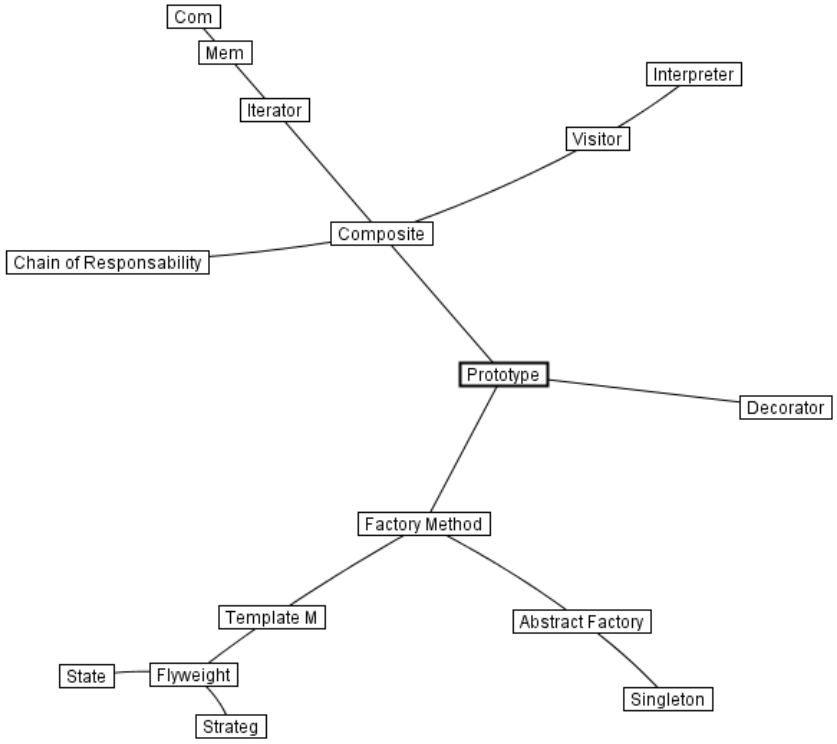

Figura 8: Árvore Hiperbólica com Padrões de Projeto

Na APRI existia apenas o componente *Web Service*, na arquitetura APRImora este componente está dividido em dois componentes, o *Web Service* e o *Semantic Web Service*. O *Web Service* disponibiliza os seguintes serviços:

- *Portrayal Service* - serviços que suportam a visualização dos diagramas das soluções propostas pelos padrões;
- *Catalog Service* - serviço que possibilita a descoberta e utilização de padrões e de serviços, com base nos seus metadados descritores;
- *Access Service* - serviços que permitem acessar e fazer download dos padrões;
- *Collaboration Service* - serviços que permitem projetistas e desenvolvedores compartilharem suas experiências de uso para aperfeiçoar os padrões;
- *MDA Service* - serviços que permitem a geração de código fonte a partir de diagramas;
- *Validation Service* - serviço que permite a validação de metadados.

O componente *Semantic Web Service* disponibiliza os seguintes serviços:

- *RDF Service* - serviço que permite a geração de documentos no formato RDF;
- *Hyperbolic Tree* – serviço que gera uma visualização das ligações existentes entre os padrões e permite a navegação entre eles.

Os componentes *Web Service* e o *Semantic Web Service* possuem uma ligação de via dupla com o *Linked Open Data* indicando que os serviços podem realizar buscas externamente, desde que o resultado obtido esteja em um formato compatível com os suportados pela busca realizada pela APRImora. Ou seja, tanto internamente a arquitetura como externamente, a linguagem utilizada para recuperação de informação será o SparQL, e a busca será feita apenas em arquivos no formato RDF.

A ligação existente entre *Web Service*/*Semantic Web Service* e o *Linked Open Data* implica na possibilidade de recuperação e descoberta de outras fontes de dados que estejam relacionadas aos padrões, sempre que estes forem publicados. E o contrário também é possível, pois agentes externos podem realizar buscas nos repositórios da APRImora utilizando serviços.

5 DC2DP METADATA EDITOR

O DC2DP *Metadata Editor*, citado na seção 4.1, é um sistema Web que permite ao usuário criar, editar e salvar metadados de acordo com as regras descritas no perfil de aplicação DC2DP. Com este sistema, é possível exportar os metadados para o formato de arquivo RDF. A Figura 9 mostra a página inicial do *DC2DP Metadata Editor*.

No DC2DP Metadata Editor é possível navegar entre os elementos do perfil DC2DP por meio de uma árvore de diretório (Figura 9a).

Para que o usuário edite os metadados em conformidade com as regras do DC2DP, cada elemento possui logo abaixo do seu nome (Figura 9b) um acrônimo indicando as regras de obrigatoriedade, ocorrência e tipo de valor permitidos para aquele elemento. A Tabela 1 apresenta os acrônimos destas regras.

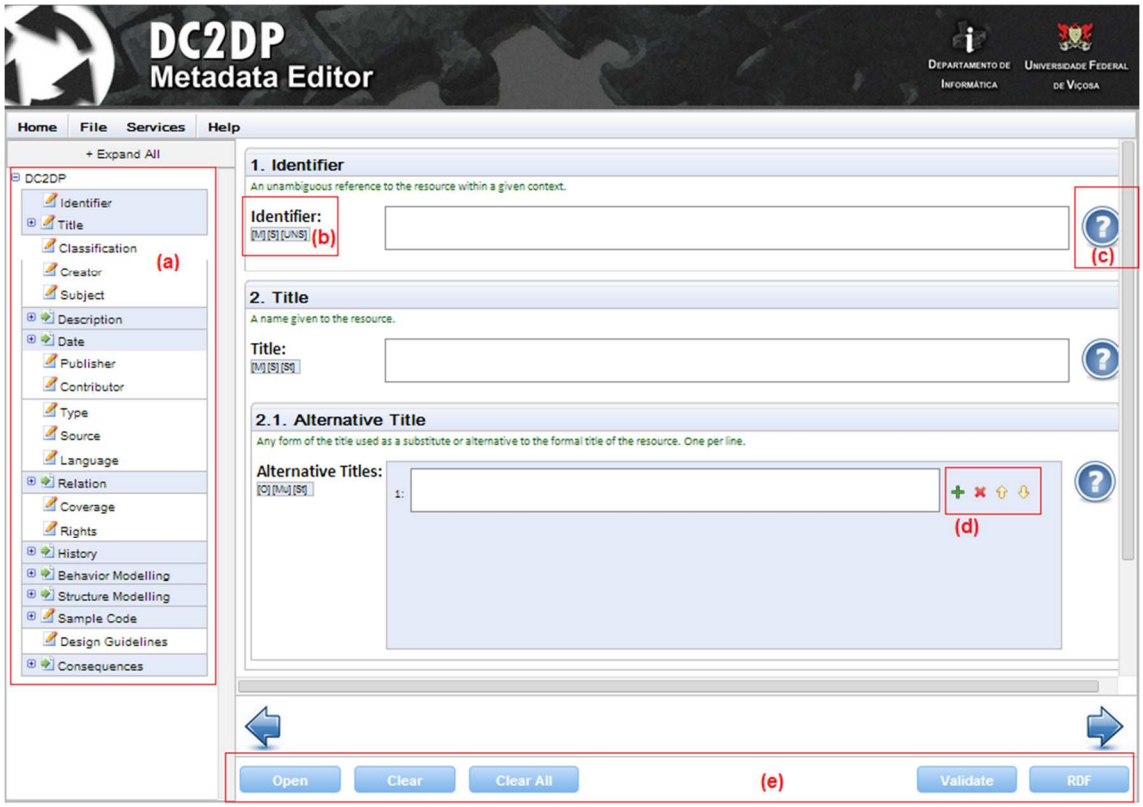

Figura 9: DC2DP Metadata Editor

Além dos acrônimos o usuário pode verificar o significado de cada elemento através da função "Ajuda" (Figura 9c). Essa opção encaminha o usuário para o documento da descrição técnica dos elementos do DC2DP, exatamente na seção correspondente ao elemento desejado, o que é feito com base em sua URI única, definida pelo PURL.

Tabela 1: Acrônimos das Regras do DC2DP

Acrônimos das Regras		
Obrigatoriedade	Ocorrência	Tipo de Valor
[M] Obrigatório [O] Opcional [Cd] Condicional	[S] Simples [Mu] Múltiplos	[St] String [D] Data [U] URI [N] Null [UNS] URI, número or string [US] URI or string

Para os elementos de ocorrência múltipla existem quatro tipos de botões (Figura 9d) onde o usuário pode inserir campos adicionais para descrever mais informações relacionadas aquele elemento ( ), excluir o campo adicionado ( ), e alterar a ordem desses campos ( ).

Uma vez que o usuário tenha editado os elementos do padrão, ele pode utilizar as funções disponibilizadas pelo editor (Figura 9e), sendo elas: *Open* - abrir padrões

documentados anteriormente e salvos no formato RDF; *Clear* - limpa os dados dos campos da página atual; *Clear all* - limpa os dados de todas as páginas; *Validate* - verifica se todos os campos tidos como obrigatórios foram preenchidos; e *RDF* - salva um documento RDF contendo os elementos do padrão documentado.

6 CONCLUSÕES E TRABALHOS FUTUROS

Um dos grandes problemas encontrados pelos desenvolvedores de software, no que diz respeito às fases de análise e de projeto, é a descoberta de padrões de reuso que possam atender as suas necessidades. A utilização de máquinas de busca para recuperar padrões nem sempre ajudam muito, uma vez que uma simples pesquisa pode retornar um número elevado de resultados e na maioria das vezes resultados que não são relevantes para o contexto da pesquisa.

A APRI foi proposta com o intuito de auxiliar os desenvolvedores nessa busca. Os padrões de análise e de projeto contidos na APRI são documentados utilizando os perfis de aplicação DC2AP e o DC2DP, respectivamente, e posteriormente armazenados em seus repositórios.

Porém, não basta apenas armazenar os padrões em repositórios, uma vez que existe centenas de padrões e a busca manual pode ser exaustiva e utilizando máquinas de busca podem retornar números elevados de resultados.

Sendo assim, o principal objetivo deste artigo foi propor a extensão da APRI para uma arquitetura capaz de auxiliar o usuário na recuperação aprimorada de padrões contidos em seus repositórios.

Para alcançar esse objetivo a APRI foi estendida para uma arquitetura que tenha como base a Web Semântica, tornando-a capaz de auxiliar o usuário na recuperação dos padrões contidos em seus repositórios, dando origem assim à APRImora, uma Infraestrutura Semântica de Reuso de Padrões.

As camadas da Web Semântica foram utilizadas para estender a APRI, uma vez que estas fornecem meios para que os dados sejam organizados de forma a serem interpretados facilmente por computadores. Além disso a forma da Web Semântica organizar os dados permite que os resultados de busca sejam melhorados.

Dessa forma, com a adição das camadas da Web Semântica na APRI, é esperado que as informações sobre os artefatos computacionais armazenados nos repositórios sejam recuperadas de forma precisa, permitindo que o usuário reutilize os padrões que melhor atendem a suas necessidades, além de evitar o desperdício de tempo com pesquisas exaustivas que não retornam resultados relevantes.

Como trabalhos futuros propõem-se a criação de um protótipo funcional da estrutura da APRImora para que experimentos possam ser feitos para comprovar a sua eficiência. Ainda como trabalhos futuros propõem-se a adição de agentes de software na estrutura da APRImora para auxiliar na recuperação de conhecimento.

AGRADECIMENTOS

Este projeto foi parcialmente financiado pela CAPES - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, por meio de bolsas de estudo e recebeu apoio financeiro das agências de fomento CNPq - Conselho Nacional de Desenvolvimento Científico e Tecnológico e FAPEMIG - Fundação de Amparo à Pesquisa do Estado de Minas Gerais.

3 CONCLUSÕES GERAIS E TRABALHOS FUTUROS

O principal objetivo desse trabalho foi estender a estrutura da APRI, propondo melhorias para que fossem obtidos, nas buscas por padrões nos repositórios da APRI, resultados precisos e que consequentemente beneficiassem a reutilização desses padrões.

Para alcançar os objetivos específicos dessa pesquisa, um mapeamento entre os elementos pertencentes ao *template* proposto por Gamma et al. (2005) e o padrão de metadados Dublin Core foi realizado. Em seguida, o mesmo processo foi realizado entre os elementos obtidos no mapeamento entre Dublin Core e Gamma e os elementos do Perfil de Aplicação Dublin Core para Padrões de Análise (DC2AP).

Essa comparação foi realizada com o intuito de verificar a existência de diferenças na forma de documentar padrões de projeto e de análise, além de averiguar se existia a necessidade da criação de um novo padrão para documentar padrões de projeto ou se era viável apenas a criação de um perfil a partir dos elementos do DC2AP. Avaliando essa comparação foi possível definir a necessidade de um novo perfil para a documentação de padrões de projeto, dando origem assim ao Perfil de Aplicação Dublin Core para Padrões de Projeto (DC2DP).

O DC2DP, proposto no Artigo I, foi criado especificamente para descrever padrões de projeto de maneira compatível com a proposta da APRI. O DC2DP fornece aos usuários informações mais completas referentes aos padrões de projetos, com isso o projetista será capaz de escolher o padrão que melhor se adequa na realização do seu projeto.

Posteriormente, a arquitetura da APRI foi estendida com a adição de um repositório de padrões de projeto, permitindo assim que os padrões de projeto possam ser mais facilmente encontrados, estudados, e reutilizados. Com isso a estrutura da APRI se torna mais abrangente, permitindo que a reutilização do conhecimento não só seja realizada durante a etapa de análise de desenvolvimento de software, mas também na etapa de projeto.

Porém, com base nos estudos e pesquisas realizadas foi possível notar que apenas armazenar os padrões em repositórios não é suficiente, é preciso disponibilizar meios para que as buscas realizadas nesses repositórios obtenham resultados precisos, retornando em cada busca, padrões que correspondam realmente àquilo que o projetista está buscando.

Dessa forma, o objetivo de alcançar melhorias nas buscas por padrões nos repositórios da APRI, obtendo resultados precisos e que consequentemente beneficiem a reutilização desses padrões foi alcançado através da proposta de adição de elementos semânticos à estrutura da APRI, dando origem à APRImora, uma infraestrutura semântica de reuso de artefatos computacionais.

Os elementos da Web Semântica adicionados na APRImora, aliados ao conceito de *Linked Data* e serviços Web, foram ferramentas importantes para alcançar o objetivo desse trabalho, uma vez que com a utilização desses conceitos os artefatos computacionais podem ser facilmente recuperados e consequentemente reutilizados.

Como possíveis trabalhos futuros, pode-se apontar o estudo de formas de realização de recuperação automática de conhecimento na APRImora por meio de agentes de softwares. Ainda como trabalhos futuros propõe-se a realização de um experimento para analisar de forma quantitativa o impacto da reutilização dos artefatos recuperados na APRImora.

APÊNDICE A

Descrição semântica dos elementos do DC2DP

Elements of DC2DP		
1.Identifier		
2. Title	2.1 Alternative Title	
3. Classification		
4. Creator		
5. Subject		
6. Description	6.1 Problem	
	6.2 Motivation	6.2.1Example
		6.2.2 Known Uses
7. Date	7.1 Created	
	7.2 Modified	
8. Publisher		
9. Contributor		
10. Type		
11. Source		
12. Language		
13. Relation	13.1 Is version of	
	13.2 Is Replaced By	
	13.3 Replaces	
	13.4 Is Part of	
	13.5 Has Part	
	13.6 Related Patterns	
	13.7 Is Analyzed With	
14. Coverage		
15. Rights		
16. History	16.1 Event Date	
	16.2 Author	
	16.3 Reason	
	16.4 Changes	
17. Behavior Modelling	17.1 Sequence Diagram	
18. Structure Modelling	18.1 Object Diagram	
	18.2 Class Diagram	18.2.1 Class Description
		18.2.2Relation Description
19. Sample Code	19.1 Programming Language	
	19.2 Code	
	19.3 Code Description	
20. Design Guidelines		
21. Consequences	21.1 Positive	
	21.2 Negative	

- ***Identifier:*** Is an unambiguous reference given to a design pattern (DP) to differentiate it from others.
- ***Title:*** Formal name by which a DP is widely known. The pattern's name conveys the essence of the pattern succinctly. Thus a good name is vital, because it will become part of your design vocabulary.
- ***Alternative Title:*** Other name whereby the pattern is known, if there.
- ***Classification:*** Classifies patterns according to two criteria: the purpose and scope. The purpose reflects what the standard is, this is, its functionality. According to this

criterion, a pattern can be: creative (with respect to the process of creating objects), structural (deals with the composition of classes or objects) or behavioral (characterizes the ways in which classes or objects interact and distribute responsibility). The scope on the other hand turn specifies whether the pattern is centered classes (and in this case makes extensive use of inheritance) or objects (and therefore more based on associations).

- ***Creator:*** Creator's name of a DP.
- ***Subject:*** Are keywords related to the scope in which a DP is applied, thus providing a brief contextualization.
- ***Description:*** Agglutinating element of descriptions aimed in the contextualization of a DP over various aspects.
- ***Problem:*** Brief textual description of what the pattern do and what the problem solved by a design pattern.
- ***Motivation:*** Describes a scenario that illustrates a design problem and how the class and object structures in the pattern solves the problem.
- ***Examples:*** Examples of situations in which the design pattern can be applied, examples of poor designs that pattern can address, and how can recognize these situations.
- ***Know Use:*** Known uses of the documented design pattern in real systems and applications.
- ***Date:*** Agglutinating element of dates about the creation and modification of the documented design pattern.
- ***Created:*** Creation date of the version of the documented DP.
- ***Modified:*** Date of the last modification of the documented DP.
- ***Publisher:*** Names of responsible for providing the DP for the public.
- ***Contributor:*** Names of responsible for changes made in the original version of the design pattern.
- ***Type:*** The nature of documented resource. It should always be Design Pattern.
- ***Source:*** Reference to the DP used as the main intellectual source in the creation of a DP.
- ***Language:*** Language used for documenting the DP.
- ***Relation:*** Agglutinating element of existing relationships between the documented DP and other patterns.

- ***Is Version Of:*** Reference to the first version of the documented DP.
- ***Is Replaced By:*** References for all the most current versions of the documented DP.
- ***Replaces:*** References for all previous versions of the documented DP.
- ***Is Part Of:*** References for designs patterns that contain the documented DP as part of its composition.
- ***Has Part:*** References for DP which is contained as part of the composition of the documented DP.
- ***Related Patterns:*** References for DP which are related to this one documented.
- ***Is Analyzed With:*** Indicates which analysis patterns can be developed using this DP.
- ***Coverage:*** Spatial locations or time period covered by the scope of a DP. It can compensate regionalisms that generate semantic ambiguities.
- ***Rights:*** Existing rights over the documented DP.
- ***History:*** Agglutinating element of historical data about the evolution of the documented DP.
- ***Event Date:*** Date of the historic event occurrence.
- ***Author:*** Names of responsible for the occurrence of the historic event.
- ***Reason:*** Reason for the occurrence of the historical event.
- ***Changes:*** Changes made in the design pattern during an event of changes.
- ***Behavior Modelling:*** Agglutinating element of the diagrams that make up the behavioral model of the design pattern.
- ***Sequence Diagram:*** Show the flow of requests between objects.
- ***Structure Modeling:*** Agglutinating element of the graphical representation of the class in the pattern and descriptions that make up the structural model of the DP.
- ***Object Diagram:*** Creates scenarios that depict a particular object structure at run-time.
- ***Class Diagram:*** Depicts classes, their structure, and the static relationships between them.
- ***Class Description:*** Brief description of each class of the class diagram.
- ***Relation Description:*** Brief description of the main relationship present in the class diagram.
- ***Sample Code:*** Agglutinating element examples of the code fragments to illustrate as design pattern might be implemented.

- ***Programming Language:*** Name of the programming language used to illustrate the examples of design pattern.
- ***Code:*** Code fragments to illustrate as design pattern might be implemented.
- ***Code Description:*** Brief description textual about the code fragment.
- ***Designed Guidelines:*** This element presents general tips for implementation of the documented design pattern.
- ***Consequences:*** Agglutinating element of description of advantages and disadvantages of use the documented design pattern.
- ***Positive:*** Positive consequences of use of the documented design pattern.
- ***Negative:*** Negative consequences of use of the documented design pattern.

APÊNDICE B

Technical Description of Dublin Core Application Profile to Design Patterns

Angélica Aparecida de Almeida Ribeiro

Disponível em: http://www.dpi.ufv.br/projetos/apri/?page_id=566

RESUMO

Esse documento descreve um perfil de aplicação para descrição de padrões de projeto desenvolvido pelo grupo de pesquisa em Sistema de Informação da Universidade Federal de Viçosa. O formato desse documento baseou-se no trabalho de Coyle and Baker (2009) e Vegi (2012).

Palavras-chave: Dublin Core, Perfil de Aplicação, Padrões de Metadados, Padrões de Projeto.

ABSTRACT

This document describes an application profile for design patterns description developed by the Information Systems Group of Federal University of Viçosa. Its format was based in the work of Coyle and Baker (2009) and Vegi (2012).

Keywords: Dublin Core, Application Profile, Metadata Standards, Design Patterns.

I. INTRODUCTION

The DC2DP is a Dublin Core Application Profile to design patterns description. This Dublin Core Application Profile was developed by the Information Systems Group of Federal University of Vicosa based on the template proposed by Gamma et al. (1995) to specify design patterns. The main objectives of DC2DP ate to improve the retrieving and reuse of design patterns through a more precise way to be treated by a computer, thus offering important information missing before.

II. A NOTE ON DUBLIN CORE APPLICATION PROFILES

A Dublin Core Application Profile (DCAP) specifies how the Dublin Core standard is used to specify some class in a specific domain or community. A DCAP describes:

1. The set of terms used in a class of DC metadata description.
2. How the terms in this set are deployed in this class description. This includes:
    - The types of resources described by descriptions within these description sets;
    - The properties referenced in statements in those descriptions, and how those properties are used to describe resources of the specified type;
    - Requirements for the occurrence of statements using a specified property;
    - Constraints on the sets of values which are referenced in a statement using a specified property (vocabulary encoding schemes); and
    - Constraints on the datatypes of the value strings occurring in a statement using a specified property (syntax encoding schemes).

The terms description set, description, property, value, vocabulary encoding scheme, value representation, value string, syntax encoding, and related description are used in the same they are used in DCMI Abstract Model (DCAM)(Powel et al., 2007).

This document is not a description of an RDF/XML format. There may be multiple bindings of this DCAP, to RDF/XML and to other syntaxes.

III. VOCABULARIES/NAMESPACES USED IN THIS DCAP

All references to properties and classes in DC metadata descriptions are made using URI. In this document, Qualified Names of the form prefix ":" local-part are sometimes used as abbreviations for URI which identify metadata terms. Prefixes are assumed to be associated

with Namespace Names (URI) as follows, and the corresponding URI for the term is constructed by concatenating the Namespace Name and the local-part:

Vocabulary Title	Namespace name	Prefix
DC2DP Element Set	http://purl.org/dc2dp/elements/	**dc2dp**
DC2AP Element Set	http://purl.org/dc2ap/elements/	**dc2ap**
DC2DP Vocabulary Encoding Schemes	http://purl.org/dc2dp/ves/	**dc2dpves**
DC2DP Type Vocabulary	http://purl.org/dc2dp/type/	**dc2dptp**
The Dublin Core Metadata Element Set, v1.1	http://purl.org/dc/elements/1.1/	**dc**
Dublin Core Terms	http://purl.org/dc/terms/	**dcterms**
MARC 21 Elements 7XX	http://marc21rdf.info/elements/7XX	**marc21**
RDA Roles	http://rdvocab.info/roles/	**rdaroles**

IV. HOW TO READ THE TABLES IN THIS DOCUMENT

The section "Describing a Design Pattern" describes how a set of terms (properties, classes, vocabulary encoding schemes, syntax encoding schemes) is used to construct a DC metadata description of a resource of the specified type.

A. Class

This section describes how the class, the type of resource, to which the metadata description applies is defined.

- **Class URI:** The URI by which the class is referenced in a DC metadata description.
- **Qualified Name for Class:** The Qualified Name which is typically used as an abbreviation for the class URI.
- **Defined By:** The name and identifier of the metadata vocabulary from which the class is drawn.
- **Type of Term:** An indication of the type of the term, according to the typology of the DCMI Abstract Model.
- **Subclass Off:** Class of which the current class is a subclass.
- **Label:** The short label provided for the class by its owner/maintenance agency.
- **Definition:** The definition provided for the class by its owner/maintenance agency.

- **Comments:** Additional information about the class provided by its owner/maintenance agency.

B. Property

This section describes how a specified property is used in a statement within a DC metadata description. The use of the property is described using the following attributes:

- **Property URI:** The URI by which the property is referenced in a DC metadata description.
- **Qualified Name for Property:** A unique name/identifier for the property. It is presented as a Qualified Name, but is an abbreviation for the property URI.
- **Defined By:** The name and identifier of the metadata vocabulary from which the property is draw.
- **Type of Term:** An indication of the type of the term, according to the typology of the DCMI Abstract Model.
- **Subproperty Of:** A property of which the current property is a subproperty.
- **Source Label:** The short label provided for the property by its owner/maintenance agency.
- **Label is this DCAP:** A short human-readable label that provides an indication of how the property is to be used in a DC metadata description of a resource of the specified type. The label does not appear in the description. It may be used to provide a descriptor for fields in displays of descriptions to human readers, but there is no requirement for display applications to use this label.
- **Source Definition:** The definition provided for the property by its owner/maintenance agency.
- **Usage in this DCAP:** A description of how the property is to be applied in a description of a resource of the specified type. This information supplements the definition of the property provided by its owner/maintenance agency.
- **Comments for this DCAP:** Additional information about the use of the property in a description of a resource of the specified type, typically on the values and their representation.
- **Uses Vocabulary Encoding Scheme:** The unique names/identifiers of vocabulary encoding schemes from which values for the property should be drawn. Names are

presented as Qualified Names, but are abbreviations for URIs. The URI is used to refer to the vocabulary encoding scheme in DC metadata descriptions. If no vocabulary encoding scheme is listed, then the DCAP does not specify a vocabulary encoding scheme from which values should be drawn. However, the definition and usage of the property may determine that values of only certain types are appropriate. For example, the value of the dc:creator property must be an entity capable of action.

- **Value URI:** An indication of whether, if a statement using the property (and vocabulary encoding scheme, where specified) is present, a value URI is to be used. Mandatory – a value URI is required; Mandatory, Fixed – a specified value URI is required; Optional – a value URI is optional[2]; Not permitted – a value URI is not permitted.
- **Value String:** An indication of whether, if a statement using the property (and vocabulary encoding scheme, where specified) is present, a value string is to be used. Mandatory – a value string is required; Mandatory, Fixed – a specified value string is required; Optional – a value string is optional[1]; Not permitted – a value string is not permitted.
- **Syntax Encoding Scheme(s):** The unique names/identifiers of datatypes from which value strings for the property should be drawn. Names are presented as Qualified Names, but are abbreviations for URIs. The URI is used to refer to the datatype in DC metadata descriptions. If no datatype is listed, then the DCAP does not specify a datatype from which value strings should be drawn.
- **Rich Representation:** As indication of whether, is a statement using the property (and vocabulary encoding scheme, where specified) is present, a rich representation is to be used. Mandatory – a rich representation is required; Optional – a rich representation is optional[1]; Not permitted – a rich representation is not permitted.
- **Obligation:** An indication of whether a statement using this property is required in a DC metadata description. Mandatory – a statement using this property is required; Conditional - a statement using this property is mandatory if exists the corresponding information; Optional – a statement using this property is optional.

[2] For each value, at least one of the following components must be present: a value URI, a rich representation, a value string or a (related) description.

- **Condition:** Information on any additional conditions on the obligation to include a statement referencing the property in a description of the specified type.
- **Minimum Occurrence:** The minimum number of statements referencing this property that can occur in a description of a resource of the specified type.
- **Maximum Occurrence:** The maximum number of statements referencing this property that can occur in a description of a resource of the specified type.

C. Vocabulary Encoding Scheme

This section describe the vocabulary encoding schemes referenced above is described.

- **Vocabulary Encoding Scheme URI:** The URI by which the vocabulary encoding scheme is referenced in a DC metadata description.
- **Qualified Name for Vocabulary Encoding Scheme:** The Qualified Name which is typically used as an abbreviation for the class URI.
- **Defined By:** The name and identifier of the metadata vocabulary from which the vocabulary encoding scheme is drawn.
- **Type of Term:** An indication of the type of the term, according to the typology of the DCMI Abstract Model.
- **Label:** The short label provided for the vocabulary encoding scheme by its owner/maintenance agency.
- **Definition:** The definition provided for the vocabulary encoding scheme by its owner/maintenance agency.
- **Comments for this DCAP:** Additional information about the use of the vocabulary encoding scheme in this DCAP.
- **See also:** A resource which provides further information about the vocabulary encoding scheme.
- **Used as Vocabulary Encoding Scheme For:** The type of resource and the property for which the vocabulary encoding scheme provides values.

D. Syntax Encoding Scheme

This section describe the syntax encoding schemes referenced above is described.

- **Syntax Encoding Scheme URI:** The URI by which the syntax encoding scheme is referenced in a DC metadata description.

- **Qualified Name for Syntax Encoding Scheme:** The Qualified Name which is typically used as an abbreviation for the class URI.
- **Defined By:** The name and identifier of the metadata vocabulary from which the syntax encoding scheme is drawn.
- **Type of Term:** An indication of the type of the term, according to the typology of the DCMI Abstract Model.
- **Label:** The short label provided for the syntax encoding scheme by its owner/maintenance agency.
- **Definition:** The definition provided for the syntax encoding scheme by its owner/maintenance agency.
- **Comments for this DCAP:** Additional information about the use of the syntax encoding scheme in this DCAP.
- **See Also:** A resource which provides further information about the syntax encoding scheme.
- **Used as Syntax Encoding Scheme For:** The type of resource and the property for which the syntax scheme provides value strings.

V. DESCRIBING A DESIGN PATTERN

A. Class description

Design Pattern [dc2dptp:designPattern]

Class URI	http://purl.org/dc2dp/type/designPattern
Qualified Name for Class	dc2dptp:designPattern
Defined By	DC2DP Type Vocabulary http://purl.org/dc2dp/type
Type of Term	Class
SubClass of	[n/a]
Label	Design Pattern
Definition	The intent of the standard design is to encapsulate specialist knowledge and provide it to be reused to solve a recurring problem in particular during the development of object-oriented software.
Comments	Is a computational artifact of reuse for the development of object-oriented software.

B. Properties descriptions

Identifier [dc:identifier]

Property URI	http://purl.org/dc/elements/1.1/identifier
Qualified Name for Property	dc:identifier
Defined By	Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/
Type of Term	Property
Subproperty Of	[n/a]
Source Label	Identifier
Label in this DCAP	[n/a]
Source Definition	An unambiguous reference to the resource within a given context.
Usage in this DCAP	Is a unique reference given to a design patterns to differentiate it from others, so it is necessary that every design patterns has identification.
Comments for this DCAP	From the identifier must be possible to recover the version of the design pattern documented. The identifier must be a number, string or URI.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Optional; Value String: Mandatory; Syntax encoding Scheme(s): [dcterms:URI]; Rich Represent: Not permitted
Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	1

Title [dc:title]

Property URI	http://purl.org/dc/elements/1.1/title
Qualified Name for Property	dc:title
Defined By	Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/
Type of Term	Property
Subproperty Of	[n/a]
Source Label	Title
Label in this DCAP	[n/a]
Source Definition	A name given to the resource.
Usage in this DCAP	Formal name by which a design patterns is widely known.
Comments for this DCAP	This name should be generic, allowing for adjustments for different domains of the same problem covered by the pattern, must be a string.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Not permitted; Value String: Mandatory; Syntax Encoding Scheme(s): ; Rich Represent: Not permitted
Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	1

Alternative Title [dcterms:alternative]

Property URI	http://purl.org/dc/terms/alternative
Qualified Name for Property	dcterms:alternative
Defined By	Dublin Core Terms http://purl.org/dc/terms/
Type of Term	Property
Subproperty Of	[dc:title] Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/title
Source Label	Alternative Title
Label in this DCAP	[n/a]
Source Definition	Any form of the title used as a substitute or alternative to the formal title of the resource.
Usage in this DCAP	Alternative names are used to inform other names by which the design pattern is well known, if any.
Comments for this DCAP	Different alternative names can be registered, if any. The alternative title must be a string.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Not permitted; Value String: Mandatory; Syntax Encoding Scheme(s): ; Rich Represent: Not permitted
Obligation	Optional
Condition	Design patterns don't always have alternative names, so don't fill this element refinement doesn't affect the retrieval of information.
Minimum Occurrences	0
Maximum Occurrences	Unbounded

Classification [dc2dp:classification]

Property URI	http://purl.org/dc2dp/elements/classification
Qualified Name for Property	[dc2dp:classification]
Defined By	DC2DP Element Set http://purl.org/dc2dp/elements/
Type of Term	Property
Subproperty Of	[n/a]
Source Label	Classification
Label in this DCAP	[n/a]
Source Definition	The classification is made according to two criteria: purpose and scope. The purpose reflects what the standard, i.e. its functionality is. According to this criterion, a pattern can be: creative, structural or behavioral. The scope turn specifies whether the pattern is centered classes or objects.
Usage in this DCAP	This element refinement was defined for this DCAP.
Comments for this DCAP	The classification must be a string.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Not permitted; Value String: Mandatory; Syntax Encoding Scheme(s): ; Rich Represent: Not permitted
Obligation	Optional
Condition	[n/a]
Minimum Occurrences	0
Maximum Occurrences	1

Creator [dc:creator]

Property URI	http://purl.org/dc/elements/1.1/creator
Qualified Name for Property	dc:creator
Defined By	Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/
Type of Term	Property
Subproperty Of	[n/a]
Source Label	Creator
Label in this DCAP	[n/a]
Source Definition	An entity primarily responsible for making the resource.
Usage in this DCAP	Creator's name of a design pattern.
Comments for this DCAP	The creator can be a person or company. Different creators can be registered, preferably using separate iterations of this element. The creator must be a string.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Not permitted Value String: Mandatory Syntax Encoding Scheme(s): Rich Represent: Not permitted
Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	Unbounded

Subject [dc:subject]

Property URI	http://purl.org/dc/elements/1.1/subject
Qualified Name for Property	dc:subject
Defined By	Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/
Type of Term	Property
Subproperty Of	[n/a]
Source Label	Subject
Label in this DCAP	[n/a]
Source Definition	The topic of the resource.
Usage in this DCAP	Are keywords related to the scope in which a design patterns is applied, thus providing a brief contextualization.
Comments for this DCAP	Different subjects can be registered, preferably using separate iterations of this element. The subject must be a string.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Not permitted Value String: Mandatory Syntax Encoding Scheme(s): Rich Represent: Not permitted
Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	Unbounded

Description [dc:description]

Property URI	http://purl.org/dc/elements/1.1/description
Qualified Name for Property	dc:description
Defined By	Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/
Type of Term	Property
Subproperty Of	[n/a]
Source Label	Description
Label in this DCAP	[n/a]
Source Definition	An account of the resource.
Usage in this DCAP	Agglutinating element of descriptions aimed in the contextualization of a design patterns over various aspects.
Comments for this DCAP	This element is not filled, because their data are concentrated in its element refinements.
Uses Vocabulary Encoding Scheme	[not specified]

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Not permitted	Not permitted		Not permitted

Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	1

Problem [dc2ap:problem]

Property URI	http://purl.org/dc2ap/elements/problem
Qualified Name for Property	dc2ap:problem
Defined By	DC2AP Element Set http://purl.org/dc2ap/elements/
Type of Term	Property
Subproperty Of	[dc:description] Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/description
Source Label	Problem
Label in this DCAP	[n/a]
Source Definition	Brief textual description about the problem solved by an analysis pattern.
Usage in this DCAP	This element refinement describes what the pattern does, and what is the problem solved by a project pattern.
Comments for this DCAP	The problem must be a string.
Uses Vocabulary Encoding Scheme	[not specified]

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Not permitted	Mandatory		Not permitted

Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	1

Motivation [dc2ap:motivation]

Property URI	http://purl.org/dc2ap/elements/motivation
Qualified Name for Property	dc2ap:motivation
Defined By	DC2AP Element Set http://purl.org/dc2ap/elements/
Type of Term	Property
Subproperty Of	[dc:description] Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/description
Source Label	Motivation
Label in this DCAP	[n/a]
Source Definition	Specific reasons that justify the existence of the documented analysis pattern. This refinement for element documents peculiarities of the problem which should be solved by the analysis pattern, regardless of application domain.
Usage in this DCAP	This elements refinement describes a scenario that illustrates a design problem and how the class and object structures in the pattern solves the problem. The scenario will help understand the more abstract description of the pattern that follows.
Comments for this DCAP	Different motivations can be registered, preferably using separate iterations of this element refinement. The motivation must be a string.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Not permitted Value String: Mandatory Syntax Encoding Scheme(s): Rich Represent: Not permitted
Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	Unbounded

Example [dc2ap:example]

Property URI	http://purl.org/dc2ap/elements/example
Qualified Name for Property	dc2ap:example
Defined By	DC2AP Element Set http://purl.org/dc2ap/elements/
Type of Term	Property
Subproperty Of	[dc2ap:motivation] DC2AP Element Set http://purl.org/dc2ap/elements/motivation
Source Label	Example
Label in this DCAP	[n/a]
Source Definition	Examples de applications where the documented analysis pattern can be applied. This element refinement is useful to facilitate the abstraction, since it seeks to illustrate the use of analysis pattern in practice.
Usage in this DCAP	This elements refinement presents examples of situations in which the design pattern can be applied, examples of poor designs that pattern can address, and how can recognize these situations.
Comments for this DCAP	Different examples can be registered, preferably using separate iterations of this element refinement. The example must be a string.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Not permitted Value String: Mandatory Syntax Encoding Scheme(s): Rich Represent: Not permitted
Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1

Maximum Occurrences	Unbounded

Known Uses [dc2ap:knowUses]

Property URI	http://purl.org/dc2ap/elements/knowUses
Qualified Name for Property	dc2ap:knowUses
Defined By	DC2AP Element Set http://purl.org/dc2ap/elements/
Type of Term	Property
Subproperty Of	[dc2ap:motivation] DC2AP Element Set http://purl.org/dc2ap/elements/motivation
Source Label	Know Uses
Label in this DCAP	[n/a]
Source Definition	Known uses of the documented analysis pattern in real systems and applications. This element refinement is intended to share experiences of using of the analysis pattern, since after its use, the responsible analyst can describe briefly the applications or system that used the pattern and how was useful, contributing so to future users.
Usage in this DCAP	This element refinement specifies the known uses of the documented design pattern in real systems and applications. This element refinement is intended to share experiences of using of the design pattern, contributing so to future users.
Comments for this DCAP	Different known uses can be registered, preferably using separate iterations of this element refinement. The known uses must be a string.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Not permitted Value String: Mandatory Syntax Encoding Scheme(s): Rich Represent: Not permitted
Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	Unbounded

Date [dc:date]

Property URI	http://purl.org/dc/elements/1.1/date
Qualified Name for Property	dc:date
Defined By	Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/
Type of Term	Property
Subproperty Of	[n/a]
Source Label	Date
Label in this DCAP	[n/a]
Source Definition	A point or period of time associated with an event in the lifecycle of the resource.
Usage in this DCAP	Agglutinating element of dates about the creation and modification of the documented design pattern.
Comments for this DCAP	This element is not filled, because their data are concentrated in its element refinements.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Not permitted Value String: Not permitted Syntax Encoding Scheme(s): Rich Represent: Not permitted
Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1

Maximum Occurrences	1

Created [dcterms:created]

Property URI	http://purl.org/dc/terms/created
Qualified Name for Property	dcterms:created
Defined By	Dublin Core Terms http://purl.org/dc/terms/
Type of Term	Property
Subproperty Of	[dc:date] Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/date
Source Label	Created
Label in this DCAP	[n/a]
Source Definition	Data of creation of the resource.
Usage in this DCAP	Creation of the version of the documented design pattern.
Comments for this DCAP	The creation date must be in accordance with a format specification of date and time.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Optional Value String: Mandatory Syntax Encoding Scheme(s): {dcterms:W3CDTF} Rich Represent: Not permitted
Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	1

Modified [dcterms:modified]

Property URI	http://purl.org/dc/terms/modified
Qualified Name for Property	dcterms:modified
Defined By	Dublin Core Terms http://purl.org/dc/terms/
Type of Term	Property
Subproperty Of	[dc:date] Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/date
Source Label	Modified
Label in this DCAP	[n/a]
Source Definition	Date on which the resource was changed.
Usage in this DCAP	Date of the last modification of the documented design pattern.
Comments for this DCAP	The modification date must be in accordance with a format specification of date and time.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Optional Value String: Mandatory Syntax Encoding Scheme(s): {dcterms:W3CDTF} Rich Represent: Not permitted
Obligation	Conditional
Condition	If the design pattern documented is the current version of the pattern, this element refinement should not be registered; otherwise it should display the creation date of the newest version immediately newer than the documented version.
Minimum Occurrences	0
Maximum Occurrences	1

Publisher [dc:publisher]

Property URI	http://purl.org/dc/elements/1.1/publisher
Qualified Name for Property	dc:publisher
Defined By	Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/
Type of Term	Property
Subproperty Of	[n/a]
Source Label	Publisher
Label in this DCAP	[n/a]
Source Definition	An entity responsible for making the resource available.
Usage in this DCAP	Names of responsible for providing the design pattern for the public. Publishers can be people or companies.
Comments for this DCAP	Different publishers can be registered, preferably using separate iterations of this element. The publisher must be a string.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Not permitted Value String: Mandatory Syntax Encoding Scheme(s): Rich Represent: Not permitted
Obligation	Optional
Condition	In situations in which the creators of the design pattern are also responsible for the availability for the public, it is not necessary to register them again as publishers.
Minimum Occurrences	0
Maximum Occurrences	Unbounded

Contributor [dc:contributor]

Property URI	http://purl.org/dc/elements/1.1/contributor
Qualified Name for Property	dc:contributor
Defined By	Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/
Type of Term	Property
Subproperty Of	[n/a]
Source Label	Contributor
Label in this DCAP	[n/a]
Source Definition	An entity responsible for making contributions to the resource.
Usage in this DCAP	Names of responsible for changes made in the original version of the design pattern. The contributors can be people or companies.
Comments for this DCAP	Different contributors can be registered, preferably using separate iterations of this element. The publisher must be a string.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Not permitted Value String: Mandatory Syntax Encoding Scheme(s): Rich Represent: Not permitted
Obligation	Conditional
Condition	This element must be registered, only in situations where the design pattern documented is a version with modifications.
Minimum Occurrences	0
Maximum Occurrences	Unbounded

Type [dc:type]

Property URI	http://purl.org/dc/elements/1.1/Type
Qualified Name for Property	dc:type
Defined By	Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/
Type of Term	Property
Subproperty Of	[n/a]
Source Label	Type
Label in this DCAP	[n/a]
Source Definition	The nature or genre of the resource.
Usage in this DCAP	The nature of documented resource. Because is an element belonging to a profile of design pattern documentation, it should always inform only the nature "Design Pattern" in the appropriate language used in the documentation.
Comments for this DCAP	The type must be in accordance with a fixed term of a Vocabulary Encoding Scheme.
Uses Vocabulary Encoding Scheme	[dc2apves:type] DC2AP Vocabulary Encoding Schemes http://purl.org/dc2ap/ves/type

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Mandatory, Fixed [dc2dptp:designPattern]	Mandatory, Fixed [designPattern]		Not permitted

Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	1

Source [dc:source]

Property URI	http://purl.org/dc/elements/1.1/source
Qualified Name for Property	dc:source
Defined By	Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/
Type of Term	Property
Subproperty Of	[n/a]
Source Label	Source
Label in this DCAP	[n/a]
Source Definition	A related resource from which the described resource is derived.
Usage in this DCAP	Reference to the design pattern used as the main intellectual source in the creation of a design pattern.
Comments for this DCAP	The design pattern documented may have been derived from part of the source or even fully. The source must be a number, string or URI.
Uses Vocabulary Encoding Scheme	[not specified]

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Optional	Mandatory	[dcterms:URI]	Not permitted

Obligation	Conditional
Condition	If the design pattern documented has not based on no other, this element should not be registered.
Minimum Occurrences	0
Maximum Occurrences	1

Language [dc:language]

Property URI	http://purl.org/dc/elements/1.1/language
Qualified Name for Property	dc:language
Defined By	Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/
Type of Term	Property
Subproperty Of	[n/a]
Source Label	Language
Label in this DCAP	[n/a]
Source Definition	A language of the resource.
Usage in this DCAP	Language used for documenting the design pattern.
Comments for this DCAP	The language must be a string and may also be provided in accordance with a fixed term of a Vocabulary Encoding Scheme.
Uses Vocabulary Encoding Scheme	[dcterms:ISO639-3] Dublin Core Terms http://purl.org/dc/terms/ISO639-3

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Optional	Mandatory		Not permitted

Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	1

Relation [dc:relation]

Property URI	http://purl.org/dc/elements/1.1/relation
Qualified Name for Property	dc:relation
Defined By	Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/
Type of Term	Property
Subproperty Of	[n/a]
Source Label	Relation
Label in this DCAP	[n/a]
Source Definition	A related resource.
Usage in this DCAP	Agglutinating element of existing relationships between the documented design pattern and other patterns.
Comments for this DCAP	This element is not filled, because their data are concentrated in its element refinements.
Uses Vocabulary Encoding Scheme	[not specified]

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Not permitted	Not permitted		Not permitted

Obligation	Conditional
Condition	If there are no relationship between design pattern documented and another pattern or this is, this element not be registered.
Minimum Occurrences	0
Maximum Occurrences	1

Is Version Of [dcterms:isVersionOf]

Property URI	http://purl.org/dc/terms/isVersionOf
Qualified Name for Property	dcterms:isVersionOf
Defined By	Dublin Core Terms http://purl.org/dc/terms/
Type of Term	Property
Subproperty Of	[dc:relation] Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/relation
Source Label	Is Version Of
Label in this DCAP	[n/a]
Source Definition	A related resource of which the described resource is a version, edition or adaptation.
Usage in this DCAP	Reference to the first version of the documented design pattern.
Comments for this DCAP	This element refinement must be a number, string or URI.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Optional; Value String: Mandatory; Syntax Encoding Scheme(s): [dcterms:URI]; Rich Represent: Not permitted
Obligation	Conditional
Condition	If the design pattern documented has not earlier versions, this element refinement should not be registered.
Minimum Occurrences	0
Maximum Occurrences	1

Is Replaced By [dcterms:isReplacedBy]

Property URI	http://purl.org/dc/terms/isReplacedBy
Qualified Name for Property	dcterms:isReplacedBy
Defined By	Dublin Core Terms http://purl.org/dc/terms/
Type of Term	Property
Subproperty Of	[dc:relation] Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/relation
Source Label	Is Replaced By
Label in this DCAP	[n/a]
Source Definition	A related resource that supplants displaces or supersedes the described resource.
Usage in this DCAP	References for all the most current versions of the documented design pattern.
Comments for this DCAP	Different references to the most current versions can be registered, preferably using separate iterations of this element refinement. This element refinement must be a number, string or URI.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Optional; Value String: Mandatory; Syntax Encoding Scheme(s): [dcterms:URI]; Rich Represent: Not permitted
Obligation	Conditional
Condition	If the design pattern documented is most current version, this element refinement should not be registered.
Minimum Occurrences	0
Maximum Occurrences	Unbounded

Replaces [dcterms:replaces]

Property URI	http://purl.org/dc/terms/replaces
Qualified Name for Property	dcterms:replaces
Defined By	Dublin Core Terms http://purl.org/dc/terms/
Type of Term	Property
Subproperty Of	[dc:relation] Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/relation
Source Label	Replaces
Label in this DCAP	[n/a]
Source Definition	A related resource that is supplanted, displaced, or superseded by the described resource.
Usage in this DCAP	References for all previous versions of the documented design pattern.
Comments for this DCAP	Different references to previous current versions can be registered, preferably using separate iterations of this element refinement. This element refinement must be a number, string or URI.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Optional; Value String: Mandatory; Syntax Encoding Scheme(s): [dcterms:URI]; Rich Represent: Not permitted
Obligation	Conditional
Condition	If the design pattern documented is the first version, this element refinement should not be registered.
Minimum Occurrences	0
Maximum Occurrences	Unbounded

Is Part Of [dcterms:isPartOf]

Property URI	http://purl.org/dc/terms/isPartOf
Qualified Name for Property	dcterms:isPartOf
Defined By	Dublin Core Terms http://purl.org/dc/terms/
Type of Term	Property
Subproperty Of	[dc:relation] Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/relation
Source Label	Is Part Of
Label in this DCAP	[n/a]
Source Definition	A related resource in which the described resource is physically or logically included.
Usage in this DCAP	References for designs patterns that contain the documented design pattern as part of its composition.
Comments for this DCAP	Different references to designs patterns that contains the pattern documented can be registered, preferably using separate iterations of this element refinement. The design pattern referenced may contain the whole pattern documented in its composition or only part of it. This element refinement must be number, string or URI.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Optional; Value String: Mandatory; Syntax Encoding Scheme(s): [dcterms:URI]; RichRepresent: Not permitted
Obligation	Optional
Condition	[n/a]
Minimum Occurrences	0
Maximum Occurrences	Unbounded

Has Part [dcterms:hasPart]

Property URI	http://purl.org/dc/terms/hasPart
Qualified Name for Property	dcterms:hasPart
Defined By	Dublin Core Terms http://purl.org/dc/terms/
Type of Term	Property
Subproperty Of	[dc:relation] Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/relation
Source Label	Has Part
Label in this DCAP	[n/a]
Source Definition	A related resource that is included either physically or logically in the described resource.
Usage in this DCAP	References for designs patterns which are contained as part of the composition of the documented design pattern.
Comments for this DCAP	Different references for designs patterns which are contained by the design pattern documented can be registered, preferably using separate iterations of this element refinement. The designs patterns referenced can be contained entirely or partially in the composition of the pattern documented. This element refinement must be number, string or URI.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Optional; Value String: Mandatory; Syntax Encoding Scheme(s): [dcterms:URI]; Rich Represent: Not permitted
Obligation	Optional
Condition	[n/a]
Minimum Occurrences	0
Maximum Occurrences	Unbounded

Related Patterns [dc2ap:complementedBy]

Property URI	http://purl.org/dc2ap/elements/complementedBy
Qualified Name for Property	dc2ap:complementedBy
Defined By	DC2AP Element Set http://purl.org/dc2ap/elements/
Type of Term	Property
Subproperty Of	[dc:relation] Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/relation
Source Label	Complemented By
Label in this DCAP	Related Patterns
Source Definition	Known analysis patterns can be used to complement the documented analysis pattern.
Usage in this DCAP	This element refinement is references for designs patterns which are related to this one documented.
Comments for this DCAP	Different references for designs patterns which are related by the design pattern documented can be registered, preferably using separate iterations of this element refinement. This element refinement must be number, string or URI.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Optional; Value String: Mandatory; Syntax Encoding Scheme(s): [dcterms:URI]; Rich Represent: Not permitted
Obligation	Optional
Condition	[n/a]
Minimum Occurrences	0
Maximum Occurrences	Unbounded

Is Analyzed With [dc2ap:isDesignedWith]

Property URI	http://purl.org/dc2ap/elements/isDesignedWith
Qualified Name for Property	dc2ap:isDesignedWith
Defined By	DC2AP Element Set http://purl.org/dc2dp/elements/
Type of Term	Property
Subproperty Of	[dc:relation] Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/relation
Source Label	Is Designed With
Label in this DCAP	Is Analyzed With
Source Definition	Known design patterns that can be used during the implementation of the documented analysis pattern.
Usage in this DCAP	This element refinement indicates which analysis patterns can be developed using this design pattern.
Comments for this DCAP	This element refinement is intended to share experiences of using of the design pattern, since after its use, the responsible designer can report analysis patterns used during the analysis. Different analysis patterns can be registered, preferably using separate iterations of this element refinement. This element refinement must be a number, string or URI.
Uses Vocabulary Encoding Scheme	[not specified]

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Optional	Mandatory	[dcterms:URI]	Not permitted

Obligation	Optional
Condition	[n/a]
Minimum Occurrences	0
Maximum Occurrences	Unbounded

Coverage [dc:coverage]

Property URI	http://purl.org/dc/elements/1.1/coverage
Qualified Name for Property	dc:coverage
Defined By	Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/
Type of Term	Property
Subproperty Of	[n/a]
Source Label	Coverage
Label in this DCAP	[n/a]
Source Definition	The spatial or temporal topic of the resource, the spatial applicability of the resource, or the jurisdiction under which the resource is relevant.
Usage in this DCAP	This element contains the spatial location or time period covered by the documented design pattern. Although it is an optional element not applies to all designs patterns, its use can be very useful to compensate regionalisms that generate semantic ambiguities that may adversely affect the reuse of the pattern. The description of the design pattern coverage provides the disambiguation of terms used mainly during the modeling of the pattern.
Comments for this DCAP	Different coverage can be registered, preferably using separate iterations of this element. The coverage must be a string.
Uses Vocabulary Encoding Scheme	[not specified]

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Not permitted	Mandatory		Not permitted

Obligation	Optional

Condition	[n/a]
Minimum Occurrences	0
Maximum Occurrences	Unbounded

Rights [dc:rights]

Property URI	http://purl.org/dc/elements/1.1/rights
Qualified Name for Property	dc:rights
Defined By	Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/
Type of Term	Property
Subproperty Of	[n/a]
Source Label	Rights
Label in this DCAP	[n/a]
Source Definition	Information about rights held in and over the resource.
Usage in this DCAP	Existing rights over the documented design pattern.
Comments for this DCAP	Different rights can be registered, preferably using separate iterations of this element. The right can be describe by a string or through referrals, for example, URLs (Uniform Resource Locator) for their respective approvals.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Optional; Value String: Mandatory; Syntax Encoding Scheme(s): [dcterms:URI]; Rich Represent: Not permitted
Obligation	Conditional
Condition	In the case of existence of known rights about the pattern, they must be registered no later to be transparency in the provision of the pattern.
Minimum Occurrences	0
Maximum Occurrences	Unbounded

History [dc2ap:history]

Property URI	http://purl.org/dc2ap/elements/history
Qualified Name for Property	dc2ap:history
Defined By	DC2AP Element Set http://purl.org/dc2ap/elements/
Type of Term	Property
Subproperty Of	[n/a]
Source Label	History
Label in this DCAP	[n/a]
Source Definition	Agglutinating element of historical data about the evolution of the documented analysis pattern. Historical events related to the pattern, for example, its creation or modifications made must be registered.
Usage in this DCAP	Historical events related to the pattern, its creation or modifications made must be registered, about who documented metadata.
Comments for this DCAP	Newer versions of a designs pattern should keep the history of the precious versions, adding data to it when there are new events. Each iteration of this element represents the occurrence of an event. This element is not filled, because their data are concentrated in its element refinements.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Not permitted; Value String: Not permitted; Syntax Encoding Scheme(s): ; Rich Represent: Not permitted
Obligation	Mandatory
Condition	[n/a]

Minimum Occurrences	1
Maximum Occurrences	Unbounded

Event Date [dc:date]

Property URI	http://purl.org/dc/elements/1.1/date
Qualified Name for Property	dc:date
Defined By	Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/
Type of Term	Property
Subproperty Of	[dc2ap:history] DC2AP Element Set http://purl.org/dc2ap/elements/history
Source Label	Date
Label in this DCAP	Event Date
Source Definition	A point or period of time associated with an event in the lifecycle of the resource.
Usage in this DCAP	Date of the historic event occurrence.
Comments for this DCAP	The event date must be in accordance with a format specification of date and time.
Uses Vocabulary Encoding Scheme	[not specified]

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Optional	Mandatory	[dcterms:W3CDTF]	Not permitted

Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	1

Author [rdaroles:author]

Property URI	http://rdvocab.info/roles/author
Qualified Name for Property	rdaroles:author
Defined By	RDA Roles http://rdvocab.info/roles/
Type of Term	Property
Subproperty Of	[dc2ap:history] DC2AP Element Set http://purl.org/dc2ap/elements/history
Source Label	Author
Label in this DCAP	[n/a]
Source Definition	A person, family or corporate body responsible for the creation of a work or metadata documentation.
Usage in this DCAP	Names of responsible for the occurrence of the historic event.
Comments for this DCAP	Different authors can be registered, preferably using separate iterations of this element refinement. The author must be a string.
Uses Vocabulary Encoding Scheme	[not specified]

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Not permitted	Mandatory		Not permitted

Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	Unbounded

Reason [dc2ap:reason]

Property URI	http://purl.org/dc2ap/elements/reason
Qualified Name for Property	dc2ap:reason
Defined By	DC2AP Element Set http://purl.org/dc2ap/elements/
Type of Term	Property
Subproperty Of	[dc2ap:history] DC2AP Element Set http://purl.org/dc2ap/elements/history
Source Label	Reason
Label in this DCAP	[n/a]
Source Definition	Reason for the occurrence of the historical event.
Usage in this DCAP	This element refinement defines the reason for the occurrence of the historical event, for example, the reason of his modification, creation.
Comments for this DCAP	The reason must be a string.
Uses Vocabulary Encoding Scheme	[not specified]

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Not permitted	Mandatory		Not permitted

Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	1

Changes [dc2ap:changes]

Property URI	http://purl.org/dc2ap/elements/changes
Qualified Name for Property	dc2ap:changes
Defined By	DC2AP Element Set http://purl.org/dc2ap/elements/
Type of Term	Property
Subproperty Of	[dc2ap:history] DC2AP Element Set http://purl.org/dc2ap/elements/history
Source Label	Changes
Label in this DCAP	[n/a]
Source Definition	Changes made in the analysis pattern during an event of changes.
Usage in this DCAP	This element refinement defines the changes made in the design pattern during an event of changes.
Comments for this DCAP	The change must be a string.
Uses Vocabulary Encoding Scheme	[not specified]

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Not permitted	Mandatory		Not permitted

Obligation	Conditional
Condition	This element refinement should not be registered only when historical event refers to the creation of the design pattern documented.
Minimum Occurrences	0
Maximum Occurrences	1

Behavior Modelling [dc2ap:behavior]

Property URI	http://purl.org/dc2ap/elements/behavior
Qualified Name for Property	[dc2ap:behavior]
Defined By	DC2AP Element Set http://purl.org/dc2ap/elements
Type of Term	Property
Subproperty Of	[n/a]
Source Label	Behavior
Label in this DCAP	Behavior Modeling
Source Definition	Agglutinating element of the diagrams that make up the behavioral model of the analysis pattern. The behavioral model is the basis for the structural model of the analysis pattern.
Usage in this DCAP	Agglutinating element of the diagrams that make up the behavioral model of the design pattern.
Comments for this DCAP	This element refinement is not filled, because their data are concentrated in its element refinements.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Not permitted Value String: Not permitted Syntax ncoding Scheme(s): Rich Represent: Not permitted
Obligation	Optional
Condition	[n/a]
Minimum Occurrences	0
Maximum Occurrences	1

Sequence Diagram [dc2ap:collaborationSequenceDiagrams]

Property URI	http://purl.org/dc2ap/elements/collaborationSequenceDiagrams
Qualified Name for Property	dc2ap:collaborationSequenceDiagrams
Defined By	DC2AP Element Set http://purl.org/dc2ap/elements/
Type of Term	Property
Subproperty Of	[dc2ap:behavior] DC2AP Element Set http://purl.org/dc2ap/elements/behavior
Source Label	Collaboration/Sequence Diagrams
Label in this DCAP	Sequence Diagrams
Source Definition	Collaboration and Sequence diagrams are used to create scenarios that represent the execution of each use case of the analysis pattern.
Usage in this DCAP	Sequence Diagrams are used to created scenarios that show the flow of requests between objects.
Comments for this DCAP	This diagram can be represented by an URI relative to a XMI file or image.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Optional Value String: Mandatory Syntax Encoding Scheme(s): [dcterms:URI] Rich Represent: Optional
Obligation	Optional
Condition	[n/a]
Minimum Occurrences	0
Maximum Occurrences	1

Structure Modelling [dc2ap:structure]

Property URI	http://purl.org/dc2ap/elements/structure
Qualified Name for Property	[dc2ap:structure]
Defined By	DC2AP Element Set http://purl.org/dc2ap/elements
Type of Term	Property
Subproperty Of	[n/a]
Source Label	Structure
Label in this DCAP	Structure Modeling
Source Definition	Agglutinating element of the diagram and descriptions that make up the structural model of the analysis pattern.
Usage in this DCAP	This element refinement it is composed by graphical representation of the class in the pattern.
Comments for this DCAP	This element refinement is not filled, because their data are concentrated in its element refinements.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Not permitted; Value String: Mandatory; Syntax ncoding Scheme(s): ; Rich Represent: Not permitted
Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	1

Object Diagram [dc2dp:objectDiagram]

Property URI	http://purl.org/dc2dp/elements/objectDiagram
Qualified Name for Property	dc2ap:objectDiagram
Defined By	DC2DP Element Set http://purl.org/dc2dp/elements/
Type of Term	Property
Subproperty Of	[dc2ap:structure] DC2AP Element Set http://purl.org/dc2ap/elements/structure
Source Label	Object Diagram
Label in this DCAP	[n/a]
Source Definition	Object Diagrams are used to created scenarios that depict a particular object structure at run-time.
Usage in this DCAP	This element refinement was defined for this DCAP.
Comments for this DCAP	This diagram can be represented by an URI relative to a XMI file or image.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Optional; Value String: Mandatory; Syntax Encoding Scheme(s): [dcterms:URI]; Rich Represent: Optional
Obligation	Optional
Condition	[n/a]
Minimum Occurrences	0
Maximum Occurrences	1

Class Diagram [dc2ap:classDiagram]

Property URI	http://purl.org/dc2ap/elements/classDiagram
Qualified Name for Property	dc2ap:classDiagram
Defined By	DC2AP Element Set http://purl.org/dc2ap/elements/
Type of Term	Property
Subproperty Of	[dc2ap:structure] DC2AP Element Set http://purl.org/dc2ap/elements/structure
Source Label	Class Diagram
Label in this DCAP	[n/a]
Source Definition	Class diagram is the main part of the problem solution treated by the analysis pattern. This diagram should be created according with the objects identified by the behavioral model of the analysis pattern.
Usage in this DCAP	Class Diagrams are used to created scenario that depicts classes, their structure, and the static relationships between them.
Comments for this DCAP	This diagram can be represented by an URI relative to a XMI file or image.
Uses Vocabulary Encoding Scheme	[not specified]

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Optional	Mandatory	[dcterms:URI]	Optional

Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	1

Class Description [dc2ap:classDescriptions]

Property URI	http://purl.org/dc2ap/elements/classDescriptions
Qualified Name for Property	dc2ap:classDescription
Defined By	DC2AP Element Set http://purl.org/dc2ap/elements/
Type of Term	Property
Subproperty Of	[dc2ap:classDiagram] DC2AP Element Set http://purl.org/dc2ap/elements/classDiagram
Source Label	Class Description
Label in this DCAP	[n/a]
Source Definition	Brief description of each class of the class diagram.
Usage in this DCAP	Used to describe each class of the class diagram.
Comments for this DCAP	These descriptions are important to facilitate the understanding of the diagram and consequently of the problem solution proposed by the design pattern. Different class descriptions can be registered, preferably using separate iterations of this element refinement. The class description must be a string.
Uses Vocabulary Encoding Scheme	[not specified]

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Not permitted	Mandatory		Not permitted

Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	Unbounded

Relationship Descriptions [dc2ap:relationshipDescriptions]

Property URI	http://purl.org/dc2ap/elements/relationshipDescriptions
Qualified Name for Property	dc2ap:relationshipDescriptions
Defined By	DC2AP Element Set http://purl.org/dc2ap/elements/
Type of Term	Property
Subproperty Of	[dc2ap:classDiagram] DC2AP Element Set http://purl.org/dc2ap/elements/classDiagram
Source Label	Relationship Descriptions
Label in this DCAP	[n/a]
Source Definition	Brief description of the main relationship present in the class diagram.
Usage in this DCAP	Main relationship present in the class diagram.
Comments for this DCAP	These descriptions are important to facilitate the understanding of the diagram and consequently of the problem solution proposed by the design pattern. Different relationship descriptions can be registered, preferably using separate iterations of this element refinement. The relationship description must be a string.
Uses Vocabulary Encoding Scheme	[not specified]

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Not permitted	Mandatory		Not permitted

Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	Unbounded

Sample Code [dc2dp:sampleCode]

Property URI	http://purl.org/dc2dp/elements/sampleCode
Qualified Name for Property	dc2dp:sampleCode
Defined By	DC2DP Element Set http://purl.org/dc2dp/elements/
Type of Term	Property
Subproperty Of	[n/a]
Source Label	Sample Code
Label in this DCAP	[n/a]
Source Definition	Code fragments that illustrate how the pattern might be implemented.
Usage in this DCAP	Agglutinating element examples of the code fragments to illustrate as design pattern might be implemented.
Comments for this DCAP	This element refinement is not filled, because their data are concentrated in its element refinements.
Uses Vocabulary Encoding Scheme	[not specified]

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Not permitted	Mandatory		Not permitted

Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	1

Programming Language[marc21:programmingLanguage]

Property URI	http://marc21rdf.info/elements/7XX/M753__b
Qualified Name for Property	marc21:programmingLanguage
Defined By	MARC 21 Elements 7XX http://marc21rdf.info/elements/7XX
Type of Term	Property
Subproperty Of	[dc2dp:sampleCode] DC2AP Element Set http://purl.org/dc2dp/elements/sampleCode
Source Label	Programming Language in System Details Access to Computer Files
Label in this DCAP	Programming Language
Source Definition	Name of the programming language associated with the data comprising the computer file (e.g., Pascal).
Usage in this DCAP	Name of the programming language used to illustrate the examples of design pattern.
Comments for this DCAP	The programming language must be a string.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Not permitted Value String: Mandatory Syntax Encoding Scheme(s): Rich Represent: Not permitted
Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	1

Code [dc2dp:code]

Property URI	http://purl.org/dc2dp/elements/code
Qualified Name for Property	dc2dp:code
Defined By	DC2DP Element Set http://purl.org/dc2dp/elements/
Type of Term	Property
Subproperty Of	[dc2dp:sampleCode] DC2AP Element Set http://purl.org/dc2dp/elements/sampleCode
Source Label	Code
Label in this DCAP	[n/a]
Source Definition	Code fragments to illustrate as design pattern might be implemented.
Usage in this DCAP	This element was defined for this DCAP.
Comments for this DCAP	Different code can be registered, preferably using separate iterations of this element refinement. This element must be a string.
Uses Vocabulary Encoding Scheme	[not specified] Value URI: Not permitted Value String: Mandatory Syntax Encoding Scheme(s): Rich Represent: Not permitted
Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	Unbounded

Code Description [dc:description]

Property URI	http://purl.org/dc/elements/1.1/description
Qualified Name for Property	dc:description
Defined By	Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/
Type of Term	Property
Subproperty Of	[dc2dp:sampleCode] DC2AP Element Set http://purl.org/dc2dp/elements/sampleCode
Source Label	Description
Label in this DCAP	Code Description
Source Definition	An account of the resource.
Usage in this DCAP	This element refinement is a brief description textual about the code fragment.
Comments for this DCAP	Different code description can be registered, preferably using separate iterations of this element refinement. This element must be a string.
Uses Vocabulary Encoding Scheme	[not specified]

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Not permitted	Mandatory		Not permitted

Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	Unbounded

Design Guidelines [dc2ap:designGuidelines]

Property URI	http://purl.org/dc2ap/elements/designGuidelines
Qualified Name for Property	dc2ap:designGuidelines
Defined By	DC2AP Element Set http://purl.org/dc2ap/elements/
Type of Term	Property
Subproperty Of	[n/a]
Source Label	Design Guidelines
Label in this DCAP	[n/a]
Source Definition	This element presents general tips for implementation of the documented analysis pattern.
Usage in this DCAP	This element presents general tips for implementation of the documented design pattern.
Comments for this DCAP	This element is intended to share experiences of using of the design pattern, which can benefit future users of the pattern. Moreover this element discusses pitfalls and suggestions on the implementation of the pattern, as well as technical issues and specific language. The design guideline must be a string.
Uses Vocabulary Encoding Scheme	[not specified]

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Not permitted	Mandatory		Not permitted

Obligation	Optional
Condition	[n/a]
Minimum Occurrences	0
Maximum Occurrences	Unbounded

Consequences [dc2ap:consequences]

Property URI	http://purl.org/dc2ap/elements/consequences
Qualified Name for Property	dc2ap:consequences
Defined By	DC2AP Element Set http://purl.org/dc2ap/elements/
Type of Term	Property
Subproperty Of	[n/a]
Source Label	Consequences
Label in this DCAP	[n/a]
Source Definition	Agglutinating element of description of advantages and disadvantages of use the documented analysis pattern.
Usage in this DCAP	This element refinement describes of advantages and disadvantages of use the documented analysis pattern.
Comments for this DCAP	These positive and negative consequences are related to various aspects of the pattern. This element is not filled, because their data are concentrated in its element refinements.
Uses Vocabulary Encoding Scheme	[not specified]

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Not permitted	Not permitted		Not permitted

Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	1

Positive [dc2ap:positive]

Property URI	http://purl.org/dc2ap/elements/positive
Qualified Name for Property	dc2ap:positive
Defined By	DC2AP Element Set http://purl.org/dc2ap/elements/
Type of Term	Property
Subproperty Of	[dc2ap:consequences] DC2AP Element Set http://purl.org/dc2ap/elements/consequence
Source Label	Positive
Label in this DCAP	[n/a]
Source Definition	Positive consequences of use of the documented analysis pattern.
Usage in this DCAP	This element refinement presents the positive consequences of use the documented design pattern.
Comments for this DCAP	Different positive consequences can be registered, preferably using separate iterations of this element refinement. The positive consequence must be a string.
Uses Vocabulary Encoding Scheme	[not specified]

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Not permitted	Mandatory		Not permitted

Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	Unbounded

Negative [dc2ap:negative]

Property URI	http://purl.org/dc2ap/elements/negative
Qualified Name for Property	dc2ap:negative
Defined By	DC2AP Element Set http://purl.org/dc2ap/elements/
Type of Term	Property
Subproperty Of	[dc2ap:consequences] DC2AP Element Set http://purl.org/dc2ap/elements/consequence
Source Label	Negative
Label in this DCAP	[n/a]
Source Definition	Negative consequences of use of the documented analysis pattern.
Usage in this DCAP	This element refinement presents the negative consequences of use the documented design pattern.
Comments for this DCAP	Different negative consequences can be registered, preferably using separate iterations of this element refinement. The negative consequence must be a string.
Uses Vocabulary Encoding Scheme	[not specified]
Obligation	Mandatory
Condition	[n/a]
Minimum Occurrences	1
Maximum Occurrences	Unbounded

Value URI	Value String	Syntax Encoding Scheme(s)	Rich Represent
Not permitted	Mandatory		Not permitted

C. Vocabulary Encoding Schemes descriptions

DC2DPtype [dc2dpves:type]

Vocabulary Encoding Scheme URI	http://purl.org/dc2dp/ves/type
Qualified Name	dc2dpves:type
Defined By	DC2DP Vocabulary Encoding Schemes http://purl.org/dc2dp/ves/
Type of Term	Vocabulary Encoding Scheme
Label	DC2DP Type Vocabulary
Definition	The set of classes specified by the DC2DP Type Vocabulary, used to categorize the nature or genre of the resource.
Label in this DCAP	[n/a]
See Also	http://purl.org/dc2dp/type/
Used as Vocabulary Encoding Scheme For	Design Pattern [dc2dptp:designPattern] [dc:type] Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/type

ISO639-3 [dcterms:ISO639-3]

Vocabulary Encoding Scheme URI	http://purl.org/dc/terms/ISO692-3
Qualified Name	dcterms:ISO639-3
Defined By	Dublin Core Terms http://purl.org/dc/terms/
Type of Term	Vocabulary Encoding Scheme
Label	ISO639-3
Definition	The set of three-letter codes listed in ISO639-3 for the representation of names of languages.
Label in this DCAP	[n/a]
See Also	http://www.sil.org/iso639-3/
Used as Vocabulary Encoding Scheme For	Design Pattern [dc2dptp:designPattern] [dc:language] Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/language

D. Syntax Encoding Schemes descriptions

Uniform Resource Identifier [dcterms:URI]

Syntax Encoding Scheme URI	http://purl.org/dc/terms/URI
Qualified Name	dcterms:URI
Defined By	Dublin Core Terms http://purl.org/dc/terms/
Type of Term	Syntax Encoding Scheme
Label	Uniform Resource Identifier
Definition	The set of identifier constructed according to the generic syntax for Uniform Resource Identifiers as specified by the Internet Engineering Task Force.
Label in this DCAP	[n/a]
See Also	http://www.ietf.org/rfc/rfc3986.txt
Used as Syntax Encoding Scheme For	Design Pattern [dc2aptp:designPattern] [dc:identifier] Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/identifier
Used as Syntax Encoding Scheme For	Design Pattern [dc2aptp:designPattern] [dc:source] Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/source
Used as Syntax Encoding Scheme For	Design Pattern [dc2aptp:designPattern] [dc:rights] Dublin Core Metadata Element Set, v1.1 http://purl.org/dc/elements/1.1/rights
Used as Syntax Encoding Scheme For	Design Pattern [dc2aptp:designPattern] [dc:isVersionOf] Dublin Core Terms http://purl.org/dc/terms/isVersionOf
Used as Syntax Encoding Scheme For	Design Pattern [dc2aptp:designPattern] [dc:isReplacedBy] Dublin Core Terms http://purl.org/dc/terms/isReplacedBy
Used as Syntax Encoding Scheme For	Design Pattern [dc2aptp:designPattern] [dc:replaces] Dublin Core Terms http://purl.org/dc/terms/replaces
Used as Syntax Encoding Scheme For	Design Pattern [dc2aptp:designPattern] [dc:isPartOf] Dublin Core Terms http://purl.org/dc/terms/isPartOf
Used as Syntax Encoding Scheme For	Design Pattern [dc2aptp:designPattern] [dc:hasPart] Dublin Core Terms http://purl.org/dc/terms/hasPart
Used as Syntax Encoding Scheme For	Design Pattern [dc2aptp:designPattern] [dc2ap:complementedBy] DC2AP Element Set http://purl.org/dc2ap/elements/complementedBy
Used as Syntax Encoding Scheme For	Design Pattern [dc2aptp:designPattern] [dc2ap:isDesignedWith] DC2AP Element Set http://purl.org/dc2dp/elements/isDesignedWith
Used as Syntax Encoding Scheme For	Design Pattern [dc2aptp:designPattern] [dc2ap:collaborationSequenceDiagrams] DC2AP Element Set http://purl.org/dc2ap/elements/collaborationSequenceDiagrams
Used as Syntax Encoding Scheme For	Design Pattern [dc2aptp:designPattern] [dc2dp:objectDiagram] DC2DP Element Set http://purl.org/dc2dp/elements/objectDiagram
Used as Syntax Encoding Scheme For	Design Pattern [dc2aptp:designPattern] [dc2ap:classDiagram] DC2AP Element Set http://purl.org/dc2ap/elements/classDiagram

W3C-DTF [dcterms:W3CDTF]

Syntax Encoding Scheme URI	http://purl.org/dc/terms/W3CDTF
Qualified Name	dcterms:W3CDTF
Defined By	Dublin Core Terms http://purl.org/dc/terms/
Type of Term	Syntax Encoding Scheme
Label	W3C-DTF
Definition	The set of dates and times constructed according to the W3C Date and Time Formats Specification.
Label in this DCAP	[n/a]
See Also	http://www.w3.org/TR/NOTE-datatime
Used as Syntax Encoding Scheme For	Design Patern [dc2dptp:desingPattern] [dcterms:created] Dublin Core Terms http://purl.org/dc/terms/created
Used as Syntax Encoding Scheme For	Design Patern [dc2dptp:desingPattern] [dcterms:created] Dublin Core Terms http://purl.org/dc/terms/modified
Used as Syntax Encoding Scheme For	Design Patern [dc2dptp:desingPattern] [dcterms:created] Dublin Core Metadata Elements Set, v1.1 http://purl.org/dc/elements/1.1/date

REFERÊNCIAS BIBLIOGRÁFICAS

APACHE JENA. **Getting started with Apache Jena**. Disponivel em: <https://jena.apache.org/getting_started/index.html>. Acesso em: 10 mar. 2014

ALEXANDER, C., et al. **A Pattern Language.** Oxford University Press, New York ,1977.

ALUR, D.; CRUPI, J.; MALKS. D. **Core J2EE Patterns:** Best Practices and Design Strategies. Sun Microsystems. Palo Alto, 2003.

ANTONIOU, G.; HARMELEN, F. VAN. **A Semantic Web Primer**, Second Edition, Cambridge, MIT Press, Massachusetts, 2008.

BECKETT, D.; BROEKSTRA, J. **SPARQL Query Results XML Format (Second Edition)**. 2013. Disponível em: < http://www.w3.org/TR/2013/REC-rdf-sparql-XMLres-20130321/ > Acesso em: 24 mar. 2014.

BERNERS-LEE, T. **Weaving the web:** The past, present and future of the World Wide Web by its inventor**.** Londres, Texere Publishing, 2000.

BERNERS-LEE, T., HENDLER, J., LASSILA, O. **The semantic web.** Scientific american, vol. 284(5), pp. 28-37, 2001.

BERNERS-LEE, T.; FIELDING, R.; MASINTER, L. **Uniform Resource Identifier (URI):** generic syntax. 2005. Disponível em: <http://tools.ietf.org/html/rfc3986#section-1>. Acesso em: 19 mar. 2014.

BERNERS-LEE, T. Semantic Web Concepts. 2005. Disponível em: <http://www.w3.org/2005/Talks/0517-boit-tbl>. Acesso em: 30 jun. 2014.

BERNERS-LEE, T. **Linked Data.** 2006. Disponível em:

<http://www.w3.org/DesignIssues/LinkedData.html> Acesso em: 19 mar. 2014.

BIZER, C.; HEATH, T.; BERNERS-LEE, T. Linked Data – the story so far. **International Journal on Semantic Web and Information Systems**. v.5, n.3, p. 01-22, 2009.

BOOCH, G. Apresentação. In: Gamma, E., et al. **Padrões de Projeto:** Soluções reutilizáveis de software orientado a objetos. Porto Alegre: Bookman, 2000.

COYLE, K.; BAKER, T. **Guidelines for Dublin Core Application Profiles.** 2009. [Online]. Disponível em: <http://dublincore.org/documents/profile-guidelines/>

CUNHA, D. R.B., LÓSCIO, B. F., SOUZA, D. **Linked Data: da Web de Documentos para a Web de Dados.** In: SANTANA, A. M; NETO, P. de A. dos S.; MOURA, R. S. (Organizadores). Livro Texto dos Minicursos. Sociedade Brasileira de Computação. Escola Regional de Informática Ceará, Maranhão, Piauí (ERCEMAPI), Teresina, PI. Pp. 79-99, 2011.

DAVIS JÚNIOR, C. A.; ALVES, L. L**. Local spatial data infrastructures based on a service-oriented architecture.** In: BRAZILIAN SYMPOSIUM ON GEOINFORMATICS (GeoInfo), 8, 2005, Campos do Jordão. **Proceedings...** São José dos Campos: INPE, 2005. p. 30-45.

DCMI - Dublin Core Metadata Initiative. **Dublin Core metadata element set, v.1.0:** Reference description, 1998. Disponível em: <http://www.dublincore. org/documents/1998/09/dces/>. Acessado em: 10 mar. 2014

DCMI - Dublin Core Metadata Initiative. **Dublin Core metadata element set, v.1.1:** Reference description, 1999. Disponível em: <http://www.dublincore.org/documents/1999/07/02/dces/>. Acesso em: 10 mar. 2014

DCMI – Dublin Core Metadata Initiative. **Background.** Disponível em: <http://dublincore.org/metadata-basics/>. Acesso em: 10 mar. 2014.

DOYLE, A., REED, C., HARRISON, J., REICHARDT, M. **Introduction to OGC web services.** White Paper. OGC. 2001.

FOWLER, M. **Analysis Patterns:** reusable object models. Addison-Wesley Publishing, 1997.

GAMMA, E., et al. Design patterns: elements of reusable object-oriented software. **Reading: Addison Wesley Publishing Company**, 1995.

HARRIS, S.; SEABORNE, A. **SparQL 1.1 Query Language**. 2013. Disponível em: <http://www.w3.org/TR/2013/REC-sparql11-query-20130321/> Acesso em: 24 mar. 2014

HORTÊNCIO-FILHO, F.W.B, LÓSCIO, B. F. **Web Semântica:** Conceitos e Tecnologias**.** II Escola Regional de Computação - Ceará, Maranhão e Piauí - ERCEMAPI 2009. SBC, 2009.

HÜMMELGEN, M. **Design patterns.** 1999. Disponivel em: <http://www.batebyte.pr.gov.br/modules/conteudo/conteudo.php?conteudo=1134> Acesso em: 24 fev. 2014

KLYNE, G.; CARROLL, J. J.; MCBRIDE, B. **RDF 1.1 Concepts and Abstract Syntax.** 2014. Disponível em: < http://www.w3.org/TR/rdf11-concepts/#section-triples>. Acesso em: 19 mar. 2014.

KOBASHI, N. Y. **Vocabulário controlado: Estrutura e Utilização.** 2008. Disponível em: < http://www2.enap.gov.br/rede_escolas/arquivos/vocabulario_controlado.pdf> Acesso em: 10 abril 2014.

MARTIN, D.; DOMINGUE, J.; BRODIE, M. L.; LEYMANN, F. **Semantic Web Service, Part 1.** IEEE *Intelligent Systems*. 2007.

MONTEIRO, F. S. de. **Web semântica e repositórios digitais educacionais na área de saúde: uma modelagem com foco no objetivo de aprendizagem para refinar resultados de busca.** Tese (Doutorado em Ciência da Informação), Universidade de Brasília, DF. 2013.

NISO U.S. - NATIONAL INFORMATION STANDARDS ORGANIZATION. **The Dublin Core metadata element set:** an American national standard. Bethesda, MD, USA: NISO Press, 2001.

NISO U.S. - NATIONAL INFORMATION STANDARDS ORGANIZATION. **The Dublin Core metadata element set:** an American national standard. Bethesda, MD, USA: NISO Press, 2007.

NEBERT, D. D. **Developing spatial data infrastructures:** the SDI cookbook, version 2.0. [S.l.]: GSDI-Technical Working Group. 2004. Disponível em: <http://www.gsdi.org/docs2004/Cookbook/cookbookV2.0.pdf>. Acesso em: 26 fev. 2014.

PAMPLONA, V. F. **Web Services:** Construindo, disponibilizando e acessando Web Services via J2SE e J2ME. Disponível em:

<http://javafree.uol.com.br/artigo/871485/Web-Services-Construindo-disponibilizando-e-acessando-Web-Services-via-J2SE-e-J2ME.html> Acesso em: 10 mar. 2014

PANTOQUILHO, M., RAMINHOS, R., ARAÚJO, J. **Analysis patterns specifications:** filling the gaps. In: *Viking Plop Conference,* Bergen, Norway. p. 169-180, 2003.

PEIXOTO, D. A.; VEGI, L. F. M.; LISBOA FILHO, J. **DC2AP Metadata Editor:** A Metadata Editor For An Analysis Patterns Reuse Infrastructure. In: **CAiSE'13 FORUM AT THE 25TH INTERNATIONAL CONFERENCE ON ADVANCED INFORMATION SYSTEMS ENGINEERING**, 2013, Valencia, Spain. CEUR Workshop Proceedings, 2013. v. 998. p. 138-145.

PURL. **PURL**. Disponível em: < http://purl.oclc.org/docs/index.html >. Acesso em: 18 abril 2014.

RAMALHO, R. A. DE SÁ.; VIDOTTI, S. A. B. G.; FUJITA, M. S. L. **Web Semântica:** Uma investigação sob o olhar da Ciência da Informação. DataGramaZero – Revista de Ciência da Informação vol.8 num. 6, 2007

RIBEIRO, A. A. A; LISBOA FILHO, J.; VEGI, L. F. M.; OLIVEIRA, A. P. **DC2DP:** a Dublin Core Application Profile to Design Patterns. In: **16th International Conference on Enterprise Information Systems,** 2014, Angers. Proceedings of the 16th International Conference on Enterprise Information Systems, 2014.

SOA Pattern, 2013. Available in: <http://soapatterns.org/ design_patterns/overview>.

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

VEGI, L. F. M., et al. A machine-processable Dublin Core application profile for analysis patterns to provide linked data. In: **Int. Conf. on Dublin Core and Metadata. Kuching, Sarawak, Malaysia, 2012b.**

VEGI, L. F. M., PEIXOTO, D. A., SOARES, L. S., LISBOA-FILHO, J., OLIVEIRA, A. P. An Infraestructure Oriented for Cataloging Services and Reuse of Analysis Patterns. In: **Int. Workshop On Reuse In Business Process Management (rBPM in conjunction with The 9th Int. Conf. on Business Process Management),** 2, 2011, Clermont-Ferrand, France. **Proceedings…** Berlin Heidelberg: Springer LNBIP N.100, Part 4 – Lecture Notes in Business Information Processing, 2012c. pp. 338-343

VEGI, L. F. M.; LISBOA FILHO, J.; CROMPVOETS, J.; SOARES, L. S.; BRAGA, J. L. A Dublin Core Application Profile For Documenting Analysis Patterns In A Reuse Infrastructure. In: **International Journal of Metadata, Semantics and Ontologies**, 2013 Vol.8, No.4, pp.267 – 281

W3C. **Semantic Web.** Disponível em: <http://www.w3.org/standards/semanticweb/> Acesso em: 19 mar. 2014.

W3SCHOOLS. **Web Services Tutorial**. Disponível em:

<http://www.w3schools.com/WebServices/default.asp>. Acesso em: 19 mar 2014.

ZAIDAN, F. H. **LOD – Linked Open Data – Dados Abertos Vinculados.** Disponível em: <http://itweb.com.br/blogs/lod-linked-open-data-dados-abertos-vinculados/>. Acesso em: 15 abril 2014.