

VuFind

UMA FERRAMENTA PARA RECUPERAÇÃO DA INFORMAÇÃO

Autores

Lucas Angelo da Silveira

Diego José Macêdo

Fernando Luiz Vechiato

Ingrid Torres Schiessl

Milton Shintaku

Nathalia Brito Pinheiro Silva

Ronnie Fagundes de Brito



MINISTÉRIO DA ECONOMIA
Escola Nacional de Administração
Pública

**MINISTÉRIO DA CIÊNCIA,
TECNOLOGIA, INOVAÇÕES E
COMUNICAÇÕES**
Instituto Brasileiro de Informação
em Ciência e Tecnologia

VuFind: uma ferramenta para recuperação da informação



Brasília

2019

PRESIDÊNCIA DA REPÚBLICA

Jair Messias Bolsonaro

Presidente da República

Hamilton Mourão

Vice-Presidente da República

MINISTÉRIO DA ECONOMIA

Paulo Roberto Nunes Guedes

Ministro da Economia

Marcelo Pacheco dos Guaranys

Secretário Executivo

ESCOLA NACIONAL DE ADMINISTRAÇÃO PÚBLICA

Aline Ribeiro Dantas de Teixeira Soares

Presidente

Camile Sahb

Diretora de Gestão Interna

Diogo Godinho Ramos Costa

Diretor de Pesquisa e Pós-Graduação Stricto Sensu

Diana Magalhães de Souza Coutinho

Diretora de Seleção e Formação de Carreiras

Paulo Marques

Diretor de Educação Continuada

Guilherme Almeida

Diretor de Inovação e Gestão do Conhecimento

MINISTÉRIO DA CIÊNCIA, TECNOLOGIA, INOVAÇÕES E COMUNICAÇÕES

Marcos Cesar Pontes

Ministro da Ciência, Tecnologia, Inovações E Comunicações

INSTITUTO BRASILEIRO DE INFORMAÇÃO EM CIÊNCIA E TECNOLOGIA

Cecília Leite Oliveira

Diretora

José Luis dos Santos Nascimento

Coordenador de Planejamento, Acompanhamento e Avaliação - COPAV

Lena Vania Ribeiro Pinheiro

Coordenadora de Ensino e Pesquisa, Ciência e Tecnologia da Informação - COEPPE

Reginaldo de Araújo Silva

Coordenador De Administração - COADM

Marcos Pereira Novais

Coordenador Geral de Tecnologias de Informação e Informática - CGTI

Bianca Amaro de Melo

Coordenadora Geral de Pesquisa e Manutenção de Produtos Consolidados - CGPC

Arthur Fernando Costa

Coordenador Geral de Pesquisa e Desenvolvimento de Novos Produtos - CGNP

Milton Shintaku

Coordenador de Articulação, Geração e Aplicação de Tecnologia - COTEC



MINISTÉRIO DA ECONOMIA
Escola Nacional de Administração
Pública

**MINISTÉRIO DA CIÊNCIA,
TECNOLOGIA, INOVAÇÕES E
COMUNICAÇÕES**
Instituto Brasileiro de Informação
em Ciência e Tecnologia

VuFind: uma ferramenta para recuperação da informação

Autores

Lucas Angelo da Silveira
Diego José Macêdo
Fernando Luiz Vechiato
Ingrid Torres Schiessl
Milton Shintaku
Nathalia Brito Pinheiro Silva
Ronnie Fagundes de Brito



Brasília

2019

© 2019 Instituto Brasileiro de Informação em Ciência e Tecnologia

Esta obra é licenciada sob uma licença Creative Commons - Atribuição CC BY 4.0, sendo permitida a reprodução parcial ou total, desde que mencionada a fonte.



Impresso no Brasil

Projeto de pesquisa “*Estudo para implementação de integrador de sistemas mantidos pela Biblioteca da Escola Nacional de Administração Pública.*”

Coordenador do projeto: *Milton Shintaku - Coordenador de Articulação, Geração e Aplicação de Tecnologia (Cotec/Ibict)*

Revisão Gramatical

Rozimar Gomes da Silva Ferreira

Design Gráfico, Diagramação e Ilustrações

Suprema Gráfica Editora Ltda.

Normalização

Rozimar Gomes da Silva Ferreira

Ingrid Torres Schiessl

Autores

Lucas Angelo da Silveira

Diego José Macêdo

Fernando Luiz Vechiato

Ingrid Torres Schiessl

Milton Shintaku

Nathalia Brito Pinheiro Silva

Ronnie Fagundes de Brito

Dados Internacionais de Catalogação-na-Publicação (CIP)

Bibliotecária: Ingrid Schiessl CRB1/ 3084

V986 VuFind: uma ferramenta para recuperação da informação / Lucas Angelo da Silveira et al. -- Brasília: Ibict, 2019.

120 p.

ISBN 978-85-7013-157-7

DOI: 10.18225/9788570131584

1. Sistema de entrega e descoberta. 2. Software livre. 3. Gestão de acervos. 4. Interface de busca integrada. 5. Tecnologia da Informação. I. Silveira, Lucas Angelo. II. Macêdo, Diego José. III. Vechiato, Fernando. IV. Schiessl, Ingrid Torres. V. Shintaku, Milton. VI. Silva, Nathalia Brito Pinheiro. VII. Brito, Ronnie Fagundes de. VIII. Título.

CDD 025.524

CDU 004:004

Lista de figuras

Figura 1.1 - Funcionamento do Sistema de Descoberta e Entrega (SDE).....	20
Figura 1.2 - Estrutura organizacional VuFind.....	22
Figura 2.1 - Página de configuração no VuFind.....	33
Figura 2.2 - Configuração do banco de dados no VuFind.....	34
Figura 2.3 - Após criar o banco de dados, o aviso referente ao Database deve estar ok.....	35
Figura 2.4 - Configuração do ILS.....	35
Figura 2.5 - VuFind configurado.....	36
Figura 3.1 - Arquitetura de Harvest no VuFind.....	44
Figura 6.1 - Arquitetura VuFind na ENAP.....	95
Figura 6.2 - Arquivo xml coletado a partir do Dspace ENAP.....	100
Figura 6.3 - Metadados coletados do Dspace ENAP após a normalização.....	101
Figura 6.4 - Metadados coletados no formato oai_dc a partir do Omeka ENAP.....	103
Figura 6.5 - Metadados coletados no formato oai_dc do Omeka ENAP após normalização.....	103
Figura 6.6 - Metadados coletados no formato oai_dc a partir do OJS ENAP.....	105
Figura 6.7 - Metadados no formato oai_dc coletados do OJS ENAP após normalização.....	105
Figura 6.8 - Metadados coletados no formato marc xml do Koha ENAP.....	108
Figura 6.9 - Metadados coletados no formato marc xml do Koha ENAP após normalização.....	109
Figura 6.10 - Tema do VuFind ENAP.....	110
Figura 6.11 - Apresentação do conteúdo de um item.....	111
Figura 6.12 - Coleções no VuFind ENAP.....	112
Figura 6.13 - Alterações na busca.....	113

Lista de Quadros

Quadro 2.1 - Requisitos mínimos para instalação.....	27
Quadro 3.1 - Exemplo ilustrativo do formato OAI-DC.....	45
Quadro 3.2 - Exemplo de qualificadores QDC.....	48
Quadro 3.3 - Campos ETDMS.....	49
Quadro 3.4 - Campos no Solr.....	59
Quadro 6.1 - Possíveis valores no metadado dc.type encontrados no Dspace ENAP.....	97
Quadro 6.2 - Metadados dc mapeados em seus respectivos campos no Solr.....	98
Quadro 6.3 - Possíveis valores no metadado dc.type encontrados no OmekaENAP.....	101
Quadro 6.4- Metadados no formato marc mapeados em seus respectivos campos no Solr....	106
Quadro 6.5 - Tipos de obras armazenadas no Koha.....	106

Sumário

Prefácio (Preface).....	11
Agradecimentos.....	12
Apresentação.....	13

Capítulo 1

Introdução.....	15
1.1 Ecosistema informacional das Bibliotecas.....	15
1.2 Sistemas de Descoberta e Entrega.....	16
1.3 VuFind.....	21
1.4 Protocolo OAI-PMH.....	23
1.4.1. Solicitações OAI-PMH.....	23

Capítulo 2

Instalação e configuração.....	27
2.1 Requisitos técnicos.....	27
2.2 Instalação VuFind.....	28
2.3 Instalando o VuFind via pacotes DEB.....	31
2.4 Configuração do ambiente.....	33
2.5 Instalação de múltiplas instâncias.....	36
2.5.1 Criando uma nova instância e configuração do diretório local.....	37
2.5.2 Configurando a nova instância.....	39
2.5.3 Configurar um novo núcleo Solr.....	40
2.5.4 Configure a nova instância.....	41

Capítulo 3

Indexação.....	43
3.1 Harvesting OAI-PMH.....	43
3.1.1. Coleta no VuFind.....	49
3.2 Importação no VuFind.....	53
3.2.1 Campos Acessíveis no Solr.....	53

3.2.2	Importação de Registro MARC de uma Aplicação Específica.....	59
3.2.3	Importação de Registros XML Utilizando Transformações.....	60

Capítulo 4

Adicionando Facetas.....		67
4.1	Facetamento com Registros MARC.....	67
4.1.1	Existe o Campo de Índice.....	67
4.1.2	Campo de Índice Inexistente sem Necessidade de Normalização.....	68
4.1.3	Campo de Índice Inexistente e Tradução Necessária.....	70
4.2	Facetamento com Registros XML.....	71
4.2.1	Índice Existe no Solr.....	71
4.2.2	Índice não Existe no Solr.....	72
4.3	Navegando por Ordem Alfabética.....	73
4.4	Alterando a Busca.....	73
4.5	Hierarquias e Coleções.....	74
4.6	Re-indexação de Índices.....	78

Capítulo 5

Administração.....		81
5.1	Operações Frequentes.....	81
5.1.1	Automação.....	81
5.1.2	Iniciar e Parar o Solr.....	83
5.1.3	Configurações PHP.....	85
5.2	Integração.....	86
5.3	Alterando o Número da Porta do Solr.....	88

Capítulo 6

VuFind na ENAP.....		91
6.1	Dspace.....	91
6.2	OJS.....	92
6.3	Omeka.....	92
6.4	Koha.....	93

6.5	ProQuest.....	93
6.6	Ebook Central.....	94
6.7	Modelo Arquitetural.....	94
6.8	Integrando o Dspace ao VuFind.....	97
6.9	Integrando o Omeka ao VuFind.....	101
6.10	Integrando o OJS ao VuFind.....	104
6.11	Integrando o Koha ao VuFind.....	106
6.12	Integrando o Ebook Central ao VuFind.....	109
6.13	Configurações Aplicadas ao VuFind.....	110
6.13.1	Tema Utilizado.....	110
6.13.2	Organização das Coleções.....	112
6.13.3	Configurações de Busca.....	113
	Considerações finais.....	115
	Referências.....	117

Prefácio (Preface)

VuFind, as you will find in this book, is designed to be as easy as possible to install and get running, while at the same time offering a large array of configuration options, and an architecture that allows you to override and change whatever pieces you need to alter for your local needs. This balance between ease of use and flexibility has gained the project a large following, and the community that has grown to develop and support the software spans the globe and has been consistently welcoming to and supportive of newcomers, regardless of their level of technical skill and experience.

One of my few regrets about the project is that, like many open source projects, its documentation is not as accessible as it could be. While the project boasts a fairly extensive wiki filled with useful information, it lacks the kind of thoughtfully-organized, comprehensive introduction that a book might offer. I hope that this volume will help fill that gap, providing another path to help new users make the most of the project.

Enjoy the book, enjoy VuFind, and please feel free to get in touch if you need support or have something new to contribute!

Demian Katz*
Villanova University
March 22, 2019



*I joined the VuFind team in 2009, taking over a project begun by Andrew Nagy, who had recently moved on to pursue other opportunities. The ten years that have passed since that time have been the most rewarding of my career in libraries and software development, and this is due in large part to the VuFind software and its associated community.

Agradecimentos

O presente guia não é resultado apenas do trabalho dos autores. Muitas outras pessoas nos ajudaram a construir esta obra.

Aos colegas da Escola Nacional de Administração Pública (ENAP), por meio dos funcionários Guilherme Alberto Almeida de Almeida, Maria Stela Reis e Keicielle Schmidt.

À diretora do Instituto Brasileiro de Informação em Ciência e Tecnologia (IBICT), Cecília Leite Oliveira, pelo apoio constante ao projeto, junto a Escola Nacional de Administração Pública (ENAP).

Aos amigos da Coordenação de Articulação, Geração e Aplicação de Tecnologia (COTEC), Priscila Paiva Castro, Lucas Rodrigues Costa, Jaqueline Rodrigues de Jesus, Janinne Barcellos, Mariana Lozzi, Julia Donato, Maria Carolina Gonçalves, Priscila Rodrigues dos Santos, Frederico Oliveira, Italo Brasileiro, Rebeca Moura e Rafael Gomes, pelo apoio incondicional oferecido sempre.

Aos nossos amigos do apoio aos projetos, Valeria Paiva, Claudia Aparecida Nobre Franco, Paloma Carolyn Alves Magalhães, Ivete da Silva Vieira, Renata Monteiro Rodrigues, Rafael Reis e Sabrina Silva, pela atenção constante.

Ao professor Damian Katz, da Universidade de VillaNova, pela disponibilidade de nos ajudar sempre.

Por fim, a todos os que diretamente ou indiretamente apoiaram a elaboração desta obra.

Os autores agradecem.

Apresentação

O presente livro é resultado de um projeto de pesquisa firmado entre a Escola Nacional de Administração Pública (ENAP) e o Instituto Brasileiro de Informação em Ciência e Tecnologia (IBICT). Esta parceria ocorre desde 2014, com o apoio do IBICT na criação do repositório institucional da ENAP, no âmbito de um acordo de cooperação. Desde o início, essa parceria, fortemente embasada na colaboração, tem fortalecido os laços entre o IBICT e a ENAP.

Nesse contexto, em 2018, a ENAP e IBICT firmaram um projeto de pesquisa que contemplava, entre outros pontos, a implementação de um sistema de descoberta e entrega que criasse um portal com a busca única, com todos os sistemas de informação mantidos pela biblioteca. O cenário compunha-se de bases próprias e assinadas, resultando em um ambiente complexo. Depois da avaliação do cenário, optou-se pelo Sistema de Descoberta e Entrega VuFind.

Assim, este guia faz parte do resultado da pesquisa que visou solucionar o problema de integração do ecossistema informacional da biblioteca da ENAP, para a implementação do portal do conhecimento da ENAP. Este portal oferta a ferramenta de busca única às bases de informação da escola, tanto as próprias como as assinadas. Com isso, visa à disseminação da informação da instituição aos seus usuários.

Este livro apresenta o embasamento teórico que aporta o VuFind como ferramenta para implementação de sistemas de descoberta e entrega, mantendo a linha mais tecnológica, com vistas a apoiar as equipes de informática e informação a implementarem o sistema. O VuFind é um sistema voltado para integração de sistemas de informação, portanto, apresenta características peculiares na sua implementação, contempladas neste guia.

Assim, esta obra culmina com a descrição da implementação do VuFind na ENAP, com pormenores técnicos e tecnológicos da integração e indexação de todas as bases. Com a descrição deste estudo de caso, os autores esperam contribuir e apresentar o caminho para que outras instituições possam implementar os seus sistemas de descoberta e entrega com o VuFind, de maneira a cumprir a missão do IBICT e promover o portal de conhecimento da ENAP.

Capítulo 1

Introdução

1.1. Ecossistema informacional das Bibliotecas

As bibliotecas têm passado por mudanças significativas com o desenvolvimento e a evolução das Tecnologias de Informação e Comunicação (TIC), que propiciaram o surgimento do suporte digital que, atualmente, convive com o suporte físico nessas unidades de informação. Ribeiro (2005) relata que as bibliotecas sempre mantiveram a custódia documental, a fim de preservarem o conhecimento, mas, com os avanços tecnológicos, essa atividade passa por questionamentos, visto que essas unidades de informação passam a ter um papel de gestoras da informação cujo enfoque está na disseminação e no acesso à informação.

Nesse sentido, como defendem Shintaku e Vidotti (2017), as bibliotecas assumem um novo papel, tornando-se publicizadoras da informação, principalmente da chamada primeira fonte, por meio dos diversos sistemas de informação mantidos por elas, principalmente os repositórios institucionais. Nesse sentido, assumem um papel mais ativo da disseminação da informação, transcendendo a custódia, ganhando similaridades de editora, no que tange à disponibilização de obras.

Com os repositórios institucionais, as bibliotecas expandem as suas atividades, visto que ainda precisam manter os sistemas de gestão que contemplam o gerenciamento do acervo físico. Assim, em muitos casos, o acervo físico é

registrado no sistema de gestão de biblioteca e o acervo digital é disponibilizado nos repositórios, dando à biblioteca autonomia de gerir acervos físicos e digitais.

Cabe salientar que, no Brasil, muitas bibliotecas universitárias mantêm portais de revistas da instituição, ofertando serviços de hospedagem aos periódicos. Com isso, estendem-se ainda mais os serviços ofertados pelas bibliotecas, por meio de sistemas gestão de periódicos, que oferecem funcionalidades relacionadas ao fluxo editorial, da mesma forma que torna o ambiente informacional da biblioteca mais complexo.

Nesse cenário, as bibliotecas passam a gerir alguns sistemas informatizados em um ecossistema de informação composto por muitos elementos, que se complementam em muitos aspectos, mas que possuem características bem diferentes. Além dos sistemas mantidos pela biblioteca, cabe ressaltar as bases de conteúdos digitais assinados e outros sistemas externos que podem ofertar serviços aos usuários da biblioteca.

Dessa forma, o ecossistema informacional da biblioteca é constituído de sistemas internos, mantidos pela biblioteca, e externos, com conteúdos referenciais e digitais em texto completo. Entretanto, todo o ecossistema está voltado ao atendimento do usuário, nas suas necessidades de acesso à informação.

1.2. Sistemas de Descoberta e Entrega

A complexidade apresentada no ecossistema de informação mantido pelas bibliotecas, de forma a ofertar os serviços necessários aos seus usuários, tem se transformado em um desafio, à medida que distribui as informações em vários sistemas. Nesse sentido, Tonding e Vanz (2016) revelam a necessidade de uma interface de busca única pelas bibliotecas, frente ao ecossistema de informação mantidos por essas unidades de informação.

Nesse contexto, nascem os chamados sistemas de descoberta e entrega, originalmente em inglês, *web-scale discovery systems*, que têm por finalidade agregar metadados de vários sistemas de informação de uma instituição e ofertar uma interface única de busca. Com isso, pode-se integrar todo o ecossistema informacional da biblioteca em um sistema que possibilita a recuperação da informação, independentemente de em qual sistema a informação original se encontra, visto que nos sistemas de descoberta e entrega a indexação remete ao registro no sistema fonte.

Entretanto, um Sistema de Descoberta e Entrega (SDE) é mais que uma ferramenta de busca consolidada:

O princípio básico dos serviços de descoberta é fornecer um índice único de metadados [...] previamente coletados que permita uma busca unificada substituindo a busca federada, ou seja, a busca em cada uma das bases provedoras de conteúdos, utilizada até então. Hospedados, local ou remotamente, disponibilizam uma variedade de informações que podem incluir todo tipo de conteúdo licenciado, conteúdo local e o próprio catálogo da biblioteca, tudo combinado num índice único. (PAVÃO, 2014, f. 16).

Os sistemas de descoberta e entrega surgiram por causa da insatisfação com a falta de uma ferramenta de busca unificada, o tempo necessário para pesquisa em diversas fontes e a necessidade de conhecer os detalhes de acesso e as interfaces de diferentes provedores, conforme Tonding e Vanz (2018).

Foster (2018) entende que o SDE torna os recursos informacionais onipresentes nas bibliotecas, no que tange permitir que o mesmo recurso esteja em distintas fontes de informação ao mesmo tempo.

Para Vaughan (2011), os SDE apresentaram um grande potencial para conectar os pesquisadores à diversidade de informações existentes. Logo, iniciou-se a comercialização do SDE para as instituições de ensino superior, em consequência, tornaram-se presentes na literatura relatos sobre as experiências com diversos produtos de SDE, bem como apontamentos em estudos comparativos entre sistemas e métodos de avaliação para a seleção do sistema mais adequado para cada realidade e necessidades das bibliotecas.

Os primeiros SDEs comercializados foram *OCLC WorldCat Local*, em 2007; *Serials Solutions Summon*, em 2009; *EBSCO Discovery Service*, em 2010; *Innovative Interfaces Encore Synergy* e *Ex Libris Primo Central*, em meados de 2010. (VAUGHAN, 2011). Ressalta-se que, apesar da comercialização, surgiram no mercado da informação softwares livres que possibilitam desempenho de funcionalidades similares aos sistemas proprietários, como o VuFind, abordado neste manual, desenvolvido pela *Falvey Memorial Library* da Universidade Villa Nova.

Vaughan (2011) compreende que os SDE são definidos a partir de quatro aspectos, quais sejam: **conteúdo, descoberta, entrega e flexibilidade**. Esses aspectos dizem respeito às funcionalidades essenciais para a eficácia do sistema na instituição implementada.

Os conteúdos são os metadados fornecidos que descrevem os recursos informacionais, advindos de diversas fontes de informação que estão dispersas tanto em ambientes externos e internos da biblioteca e da instituição da qual faz parte.

Esse **conteúdo** é coletado no acervo local e demais bases de dados contratadas, sendo normalizado a partir de um esquema de metadados estipulado entre os Desenvolvedores do sistema, os provedores de conteúdo, editores e agregadores por meio dos contratos comerciais. Podem ser qualificados como as bibliotecas digitais, os repositórios institucionais, as bases de dados, resumos, referenciais e, ou texto completo e portais de periódicos científicos. Essas fontes podem ser de acesso aberto ou proprietário. Com a coleta realizada, cria-se um índice central que possui uma configuração para recuperar e exibir os resultados rapidamente. (VAUGHAN, 2011).

A construção do índice central ocorre por meio da interoperabilidade, já que as bases de dados e sistemas são heterogêneos e necessitam se corresponder para a transmissão dos metadados. Dessa forma, a coleta dos metadados (*harvesting*) ocorre por meio do protocolo *Open Archives Initiative Protocol for Metadata Harvesting* (OAI-PMH) para criar um índice central com metadados, ou seja, realizando uma pré-coleta de metadados, periodicamente, dos provedores de conteúdo (editoras científicas e periódicos científicos, bases de dados, repositórios digitais e bibliotecas digitais). (HOEPPNER, 2012; NATIONAL FEDERATION OF ADVANCED INFORMATION SERVICES, 2013).

Os metadados coletados são de textos completos e resumos que preenchem o índice central por meio de processos em lote de registros, assim vinculando os dados do acervo local da biblioteca e as coleções externas proprietárias e, ou abertas. Assim, esses metadados podem ofertar acesso a diferentes conteúdos, tais como os textos completos e, ou os resumos com a finalidade de disponibilizar aos sujeitos quando realizam suas buscas (HOEPPNER, 2012; NATIONAL INFORMATION STANDARDS ORGANIZATION, 2014; VAUGHAN, 2011), facilitando, assim, o processo de descoberta.

A **Descoberta** no SDE ocorre a partir da caixa de busca com recursos de pesquisa simples e avançada, exibidos na interface do usuário, sendo semelhante à interface de busca do Google. Segundo Breeding (2015), a interface de descoberta é exibida ao usuário final com funcionalidades como a classificação por relevância, navegação facetada e recursos baseados na Web.

Por meio da descoberta, o sujeito inicia a sua busca e utiliza estratégias e funções do sistema para aperfeiçoar os seus resultados, torna-os mais

significativos de acordo com a sua necessidade informacional. A representação e a apresentação das informações nos resultados propiciam o dinamismo na busca e na encontrabilidade da informação. Além disso, inserido no contexto da Web 2.0, o SDE possibilita a participação dos sujeitos informacionais com atribuição de tags, comentários etc.

O índice central com os metadados pré-coletados permite que o SDE realize uma recuperação mais rápida deles, assim o aspecto da **Entrega** refere-se aos resultados que são exibidos por classificação de relevância na interface. Por sua vez, a interface do sujeito deve ser intuitiva e com navegação facetada que, conseqüentemente, propiciam resultados mais específicos e personalizados.

Breeding (2015) afirma que, com a grande quantidade de resultados apresentados na interface, torna-se essencial que eles correspondam aos termos expressos na consulta, aparecendo na parte superior da lista de resultados. Essas informações reunidas são organizadas de acordo com a relevância dos recursos informacionais, advinda de um algoritmo definido pelo desenvolvedor do SDE.

Nesse sentido, a classificação por relevância é fundamental para o funcionamento do SDE, conhecida em inglês como *ranking relevance*, pois ordena os resultados entregues na interface do usuário. O desenvolvedor do sistema, muitas vezes, controla como os resultados são exibidos. No entanto, a biblioteca também deve possuir o controle, decidindo como deseja que os resultados sejam ordenados, de acordo com as necessidades informacionais dos sujeitos institucionais e informacionais. Assim, essas questões sobre o controle da apresentação dos resultados devem ser negociadas de modo transparente. (BREEDING, 2015).

A **flexibilidade** refere-se ao sistema, no que tange a sua capacidade de integração com o Sistema de Gerenciamento da Biblioteca e o *Application Programming Interface* (API). Além da interface de descoberta e entrega de resultados, o SDE pode oferecer também alguns dos serviços de informação e atendimento ao usuário. Assim, fica disponível à comunidade da biblioteca de modo remoto, possibilitando a unificação das tarefas dos sujeitos, a fim de otimizar o seu tempo e aumentando a qualidade da pesquisa. (VAUGHAN, 2011). Assim, os ambientes informacionais digitais que fornecem os conteúdos ao SDE fazem parte do ecossistema informacional com integração e oferecem uma variedade de interfaces de API. (BREEDING, 2015; SHINTAKU, 2017).

Como o Sistema de Gerenciamento da Biblioteca possui módulos como circulação, catalogação automatizada e catálogos on-line, é possível exibir o

status de um determinado item localizado na estante do acervo físico, bem como a flexibilização de reservar o item quando necessário. (BREENDING, 2015).

A integração é viabilizada pela interoperabilidade entre as APIs de provedores de conteúdo, o fornecedor do sistema da biblioteca e desenvolvedor do SDE. Com a conformidade estabelecida entre eles, através de acordos comerciais, torna-se viável o acesso unificado do sujeito aos conteúdos recuperados nas plataformas em que estão hospedados após a descoberta. A biblioteca se beneficia desses acordos e obtém autonomia em configurar o acesso e como deseja exibir os resultados. (BREENDING, 2015).

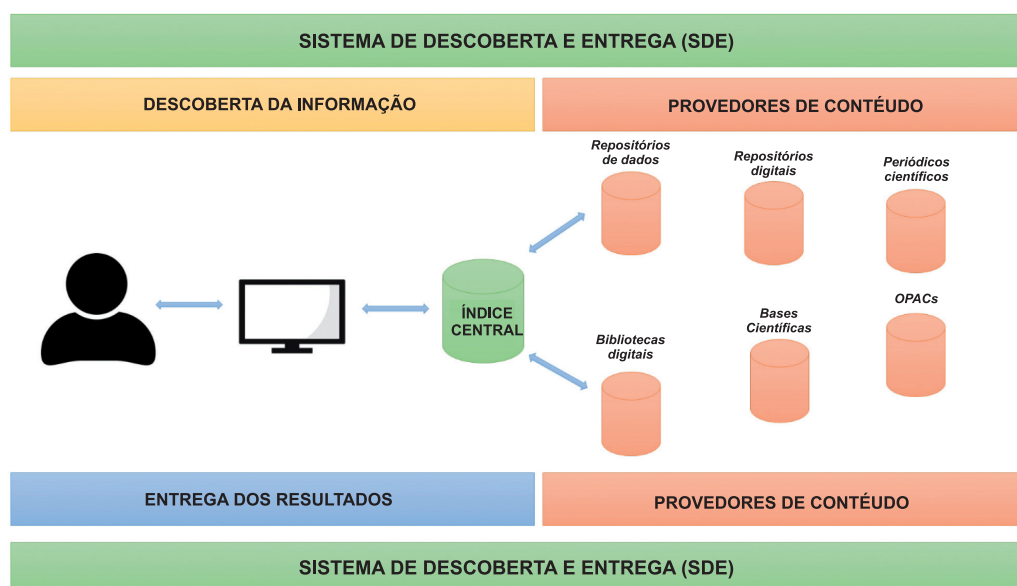


Figura 1.1 - Funcionamento do Sistema de Descoberta e Entrega (SDE).

Fonte: Elaboração dos autores (2019).

Entende-se que o processo de descoberta de informação no SDE ocorre a partir do sujeito que realiza sua pesquisa na interface, onde a sua expressão de busca é enviada ao índice central, que entrega ao sujeito resultados correspondentes à busca. Essa situação de descoberta e entrega é bilateral, ou seja, pode acontecer quantas vezes forem necessárias para que o sujeito possa encontrar o que deseja.

Além disso, o SDE precisa estar aliado com os provedores de conteúdo, quanto a possuir acordos comerciais que promovam a transparência do funcionamento do sistema e a gestão do conteúdo, propiciando a cooperação entres eles e, conseqüentemente, a conformidade com as recomendações específicas do SDE.

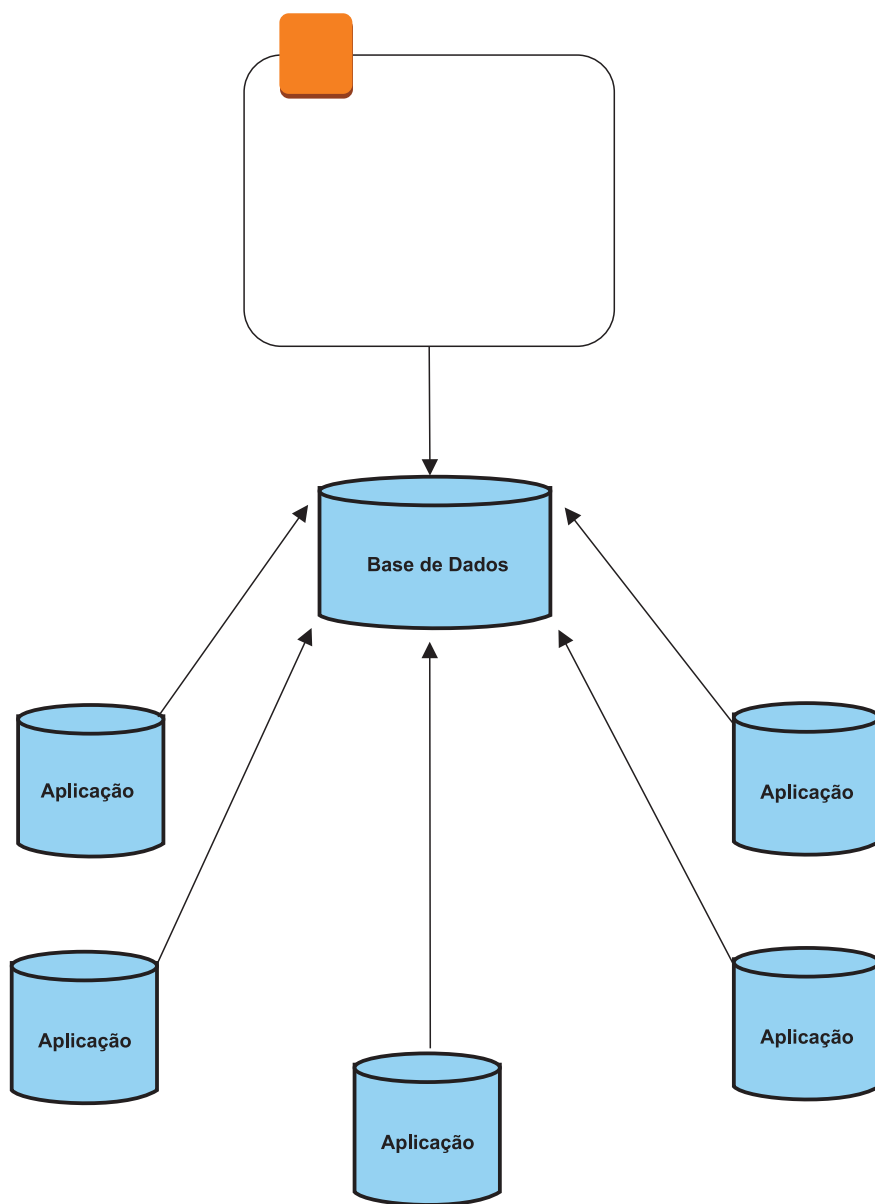
1.3. VuFind

Ferramenta de descoberta e entrega mantida pela biblioteca Falvey Memorial Library, desenvolvida na Universidade Villa Nova, Pensilvânia, Estados Unidos.

O objetivo do VuFind é permitir que seus usuários pesquisem e naveguem por todos os recursos da sua biblioteca, substituindo catálogos de acesso público online tradicionais por uma gama de serviços que incluem: registros de catálogo; itens da Biblioteca Digital, bibliografia Institucional e outras coleções de biblioteca e recursos. Além disso, o VuFind é completamente modular, ou seja, pode-se implementar tanto o sistema básico quanto um sistema completo, composto por todos os componentes, uma vez que se trata de uma ferramenta *open source* composta por uma ampla gama de opções configuráveis, permitindo personalização sem alterar nenhum código, além de permitir modificar ou adicionar módulos para melhor atender às necessidades dos usuários.

Voltado ao conjunto utilitário da ferramenta, verifica-se a capacidade de integrar sistemas gerenciadores de bibliotecas (catálogos online), bibliotecas digitais, repositórios institucionais, portais de periódicos, portais de eventos, bem como outras fontes de informações que disponibilizem os metadados para consulta, entre outras fontes abertas de informação (Na Figura 1.2 mostra-se a estrutura do VuFind). Vale ressaltar que a integração é feita via protocolo *Open Archives Initiative-Protocol for Metadata Harvesting* (OAI-PMH) que será discutido, posteriormente, no Capítulo 3. Tal protocolo é responsável por realizar a coleta dos metadados de todos os tipos de sistemas que respondem ao OAI-PMH; dessa forma, tem-se um portal de busca único para o conjunto de sistemas integrados.

Para a recuperação das informações armazenadas, o VuFind executa o *Solr Energy*. O *Apache Solr* é um mecanismo de pesquisa *open source* que oferece desempenho e escalabilidade para que o VuFind responda a consultas de busca na ordem de milissegundos, além de dispor da capacidade de ser distribuído, caso haja a necessidade e o interesse em distribuir a carga do catálogo por vários servidores.



Observação: Aplicação representa qualquer software que responda em OAI-PMH.

Figura 1.2 - Estrutura organizacional VuFind.

Fonte: Elaboração dos autores (2019).

1.4. Protocolo OAI-PMH

O protocolo *Open Archives Initiative for Metadata Harvesting* (referido como OAI-PMH) fornece uma estrutura de interoperabilidade independente da aplicação com base na coleta de metadados. Há dois tipos de participantes no framework OAI-PMH:

- Provedores de Dados responsáveis por administrar sistemas que suportam OAI-PMH como um meio de expor metadados;
- Provedores de serviços que utilizam os metadados harvested (coletados) através do OAI-PMH para disponibilizar serviços de valor agregado.

O conceito de *harvester* está ligado a um cliente que emite solicitações OAI-PMH incumbido de coletar metadados de aplicações onde uma aplicação é vista como um servidor de rede acessível que pode processar as 6 solicitações OAI-PMH: *Identifier*, *ListMetadataFormats*, *ListSets*, *ListIdentifier*, *ListRecords*, *GetRecords*. Além disso, uma aplicação é gerenciada por um provedor de dados para expor metadados aos *harvesters*. Para permitir várias configurações, o OAI-PMH distingue entre três entidades distintas relacionadas aos metadados:

- **recurso** é o objeto em que os metadados estão “sobre”. A natureza de um recurso, seja ele físico ou digital, bem como local de armazenamento no repositório está fora do escopo do OAI-PMH.
- **item** é um constituinte de um repositório a partir do qual os metadados sobre um recurso podem ser disseminados.
- **record** é o metadado em um formato específico. Um registro é retornado como um fluxo de bytes codificado em XML em resposta a uma solicitação do protocolo a um item.

As 6 solicitações OAI-PMH são discutidas brevemente a seguir.

1.4.1. Solicitações OAI-PMH

ListRecords é usado para coletar registros de uma aplicação. Argumentos opcionais permitem a coleta seletiva de registros com base na associação e, ou *datestamp*. Dependendo do suporte da aplicação para exclusões, um cabeçalho retornado pode ter um atributo de status “excluído”, caso um registro

correspondente aos argumentos especificados na solicitação já tenha sido excluído. Além disso, nenhum metadado estará presente para registros com status excluído.

GetRecord é usado para recuperar um registro de metadados individual de um repositório. Os argumentos obrigatórios especificam o identificador do item do qual o registro é solicitado e o formato dos metadados que devem ser incluídos no registro. Dependendo do nível em que um repositório rastreia exclusões, um cabeçalho com um valor “excluído” para o atributo de status pode ser retornado, caso o formato de metadados especificado pelo *metadataPrefix* (*string* para especificar o formato de metadados em pedidos OAI-PMH) não esteja mais disponível no repositório ou no item especificado.

ListIdentifier é uma forma abreviada de *ListRecords*, recuperando apenas cabeçalhos em vez de registros. Argumentos opcionais permitem a coleta seletiva de cabeçalhos com base na associação de conjunto e, ou *datestamp*. Dependendo do suporte da aplicação para exclusões, um cabeçalho retornado pode ter um atributo de status “excluído”, se um registro na solicitação tiver sido excluído.

Um *set* é uma construção opcional para agrupar itens para fins de coleta seletiva. Aplicações podem organizar itens em *sets*. A organização do conjunto pode ser plana, ou seja, uma lista simples ou hierárquica. Várias hierarquias com nodos de nível superior distintos e independentes são permitidas. Quando uma aplicação define uma organização, deve-se incluir informações de associação de *set* nos cabeçalhos dos itens retornados em resposta às solicitações *ListIdentifiers*, *ListRecords* e *GetRecord*. Dessa forma, *ListSets* é utilizado para recuperar a estrutura do conjunto de um repositório, útil para a coleta seletiva.

Identifier identifica um item dentro de um repositório; o identificador que é exclusivo é utilizado em solicitações OAI-PMH para extrair metadados do item onde se tem que todos os registros possíveis disponíveis em um único item compartilham o mesmo identificador exclusivo. Vale ressaltar que o formato do identificador exclusivo deve corresponder ao da sintaxe URI (*Uniform Resource Identifier*).

Identificadores exclusivos desempenham dois papéis no protocolo:

1. Resposta: Os identificadores são retornados pelas solicitações *ListIdentifiers* e *ListRecords*.
2. Requisição: Um identificador, em combinação com um *metadataPrefix*, é usado na solicitação *GetRecord* como um meio de solicitar um registro em um formato de metadados específico de um item.

List Metadata Formats é usado para recuperar os formatos de metadados disponíveis em um repositório. Um argumento opcional restringe a solicitação aos formatos disponíveis para um item específico. *Identifier* é um opcional argumento que especifica o identificador exclusivo do item para o qual os formatos de metadados disponíveis estão sendo solicitados. Se esse argumento for omitido, a resposta incluirá todos os formatos de metadados suportados pela aplicação. Ressalta-se que o fato de um formato de metadados ser suportado por uma aplicação não significa que ele possa ser disseminado a partir de todos os itens na aplicação.

Capítulo 2

Instalação e configuração

Neste capítulo será abordado o procedimento de instalação e configuração do software VuFind. Serão abordados os critérios técnicos, os processos de instalação, visto que é possível proceder à instalação por linha de comando ou pacotes; e ainda como instalar múltiplas instâncias. Serão também abordados os métodos para configurar o ambiente para utilização da ferramenta.

2.1. Requisitos técnicos

Para funcionamento adequado do VuFind (Quadro 2.1), os requisitos mínimos para sua instalação, de acordo com as versões de cada pacote de softwares requeridos, são:

Versões do VuFind	Versão do Apache HTTP	Versão do PHP	Versão do Mysql	Versão do Java JDK
VuFind 1.x	2.2.12+	5.2.x	4.1+	1.5+
VuFind 2.0-2.3.1	2.2.12+	5.2.x	5.1.10+	1.7+
VuFind 2.4-3.1.x	2.2.12+	5.4+	5.1.10+	1.7+
VuFind 4.x	2.2.12+	5.6+	5.5+	1.8+
VuFind 5.0+	2.2.12+	7.0.8+	5.5+	1.8+

Quadro 2.1 - Requisitos mínimos para instalação.

Fonte: Elaboração dos autores (2019).

O VuFind é compatível com a maioria dos navegadores modernos, uma ressalva em relação ao Internet Explorer, todavia, na nova versão lançada em 2018, não apresenta problemas.

Na sequência, será descrito o passo a passo da instalação no sistema operacional Debian ou Ubuntu com uso do núcleo Linux. Vale ressaltar que os comandos executados a seguir são compatíveis com as versões 8 e 9 no Debian, e na versão 16 no Ubuntu.

2.2. Instalação VuFind

A seguir, é mostrado o passo a passo de instalação, que pode ser utilizado na versão 4, 2 e 5, respectivamente. Por padrão, o local de instalação do VuFind é no diretório **/usr/local/vufind**.

Inicialmente, abra um terminal nos respectivos sistemas operacionais.

- 1 - O primeiro passo é atualizar o sistema operacional, para se obter os últimos patches disponíveis executando os comandos abaixo:

```
$ sudo apt-get update
$ sudo apt-get dist-upgrade
```

- 2 - Após atualizar os patches, é necessário reiniciar o sistema para que as atualizações sejam efetivadas.

```
$ sudo shutdown -r now
```

- 3 - Instalar o *Apache Web Server*, que é o responsável por prover acesso a web através de navegadores.

```
$ sudo apt-get -y install apache2
```

- 4 - Faz-se necessário o módulo de escrita para o apache, que é requerido pelo VuFind.

```
$ sudo a2enmod rewrite
```

- 5 - Efetiva-se o módulo, instalando no apache, o comando abaixo:

```
$ sudo /etc/init.d/apache2 force-reload
```

- 6 - Instala-se um Sistema de Gerenciamento de Banco de Dados (SGBD), para guardar os dados inseridos e produzidos pela aplicação. O VuFind suporta 2 tipos de SGBDs de código aberto: Mysql e Postgresql. Aqui, será utilizado o Mysql

```
$ sudo apt-get -y install mysql-server
```

OBSERVAÇÃO

Durante a instalação, será requisitada uma senha para o usuário root do Mysql, lembrem-se de anotá-la, posteriormente, será necessário utilizá-la.

- 7 - O VuFind foi desenvolvido a partir da linguagem de programação PHP. Assim, será necessário instalar tanto o php quanto seus respectivos módulos, que são utilizados pelo VuFind.

```
$ sudo apt-get -y install libapache2-mod-php php-mbstring php-pear php php-dev php-gd php-intl php-json php-ldap php-mysql php-xml php-soap
```

- 8 - O VuFind, utiliza como motor de busca e indexação o Solr, que é desenvolvido utilizando-se a linguagem Java. Desta forma, faz-se necessária a instalação do ambiente Java. Deve-se atentar que, na versão 5 do VuFind, o Java requerido deve ser o 8 ou posterior. Na sequência, são mostrados os detalhes da instalação do Java 8.

```
$ sudo add-apt-repository ppa:webupd8team/java
$ sudo apt-get update
$ sudo apt-get install oracle-java8-installer
```

- 9 - Após ter instalado todos os requisitos necessários para se ter um ambiente propício à instalação do VuFind, faça o download do pacote do mesmo.

```
$ cd /tmp
$ wget https://github.com/VuFind-org/VuFind/releases/download/v5.0.1/VuFind-5.0.1.tar.gz
$ tar xzvf VuFind-5.0.1.tar.gz
```

- 10 - Para escolher o diretório onde o VuFind será instalado, neste livro, será adotado o mesmo diretório recomendado na documentação, **/usr/local**. Todavia, o leitor pode ficar à vontade para alocar num diretório distinto, por exemplo, o **www**. Por questões de padronização no restante deste documento, sempre que se tratar do diretório de instalação do VuFind, o termo referido será “VUFIND_HOME”.

```
$ sudo mv VuFind-5.0.1 /usr/local/VuFind
```

- 11 - No decorrer da instalação, será perguntado em qual local é desejável realizar a instalação do diretório local do VuFind, e qual URL será utilizada para o acesso via navegador. Atente-se a esses detalhes, pois, posteriormente, será necessário utilizar tais informações.

```
$ cd /usr/local/vufind
$ php install.php
```

- 12 - Para permitir que o apache escreva no VuFind configurações e arquivos no disco, faz-se necessário dar as seguintes permissões.

```
$ sudo chown -R www-data:www-data VUFIND_HOME/local/cache
$ sudo chown -R www-data:www-data VUFIND_HOME/local/
config $sudo mkdir VUFIND_HOME/local/cache/cli
$ sudo chmod 777 VUFIND_HOME/local/cache/cli
```

- 13 - Para acessar o VuFind, é necessário linkar o apache ao mesmo através do comando abaixo.

```
$ sudo ln -s VUFIND_HOME/local/httpd-VuFind.conf /etc/apache2/
conf-enabled/VuFind.conf
```

- 14 - Faz se necessário recarregar o apache para tornar os efeitos efetivados.

```
$ sudo /etc/init.d/apache2 reload
```

- 15 - Variáveis de ambiente precisam ser configuradas para que os scripts relacionados ao VuFind e Solr possam encontrados.

```
$ sudo sh -c 'echo export JAVA_HOME=\"/usr/lib/jvm/java-8-oracle\" > /etc/profile.d/VuFind.sh'
$ sudo sh -c 'echo export VuFind_HOME=\"/usr/local/VuFind\" >> /etc/profile.d/VuFind.sh'
$ sudo sh -c 'echo export VUFIND_HOME=\"/usr/local/VuFind/local\" >> /etc/profile.d/VuFind.sh'
```

Certifique-se de que o JAVA_HOME esteja realmente em *java-8-oracle*, caso contrário preencha com o diretório correto. Para verificar, utilize o terminal com o comando abaixo:

```
$ls /usr/lib/jvm/
```

- 16 - Efetive as variáveis de ambiente. Atente, que esteja com o mesmo usuário em que o VuFind foi instalado.

```
$ source /etc/profile.d/VuFind.sh
```

- 17 - Inicie o Solr.

```
$ cd /usr/local/VuFind
$ ./solr.sh start
```

- 18 - Configure o VuFind através do navegador, acessando o link abaixo. Atente ao nome dado na URL “IPServidor” que deve ser o IP utilizado na máquina onde o VuFind está sendo implantado.

```
http://IPServidor/VuFind/Install/Home
```

2.3. Instalando o VuFind via pacotes DEB

Outra opção de implantação é a partir do pacote DEB. Esta seção fornece instruções para se fazer isso. Ele carregará rapidamente o VuFind no diretório padrão **/usr/local/VuFind**, assumindo que as configurações locais estejam em **/usr/local/VuFind/local**.

Caso tenha utilizado a instalação do Ubuntu e do Debian indicadas nos sites oficiais, não necessitará de nenhum pacote especial durante o processo de instalação do VuFind. Todos os comandos realizados são executados através da linha de comando do terminal.

Inicialmente, é necessário que os pacotes mais recentes estejam no sistema operacional. Para isto, utilize os seguintes comandos:

```
$ sudo apt-get update
$ sudo apt-get dist-upgrade
```

Deve-se reiniciar o servidor para aplicar as atualizações, com o seguinte:

```
$ sudo shutdown -r now
```

Após reiniciado o servidor, realize o *download* do pacote DEB no link oficial do VuFind. Neste guia, será utilizada a versão 5.0.1 para demonstrar os procedimentos de instalação e configuração.

```
$ wget https://github.com/VuFind-org/VuFind/releases/download/v5.0.1/VuFind_5.0.1.deb
```

Em seguida, deve-se usar o comando **dpkg** para instalação. O **dpkg** é um gerenciador de pacotes para sistemas baseados em Debian. Ele pode instalar, remover e construir pacotes, mas, ao contrário de outros sistemas de gerenciamento de pacotes, ele não pode baixar e instalar automaticamente pacotes ou suas dependências.

```
$ sudo dpkg -i VuFind_5.0.1.deb
```

Caso não tenha todas as dependências necessárias já instaladas, o comando **dpkg** produzirá um erro. Você poderá resolver esse problema executando o seguinte comando para forçar a instalação dessas dependências:

```
$ sudo apt-get install -f
```

2.4. Configuração do ambiente

Após finalizar a instalação, deve-se acessar o ambiente de configuração, partindo do pressuposto de que a instalação tenha sido realizada localmente, pode-se acessar o VuFind através de URL <http://localhost/VuFind/Install/Home>; caso a instalação tenha sido num servidor remoto, deve-se preencher com o endereço de IP ou a URL correta. Ao acessar o endereço, tem-se o cenário da Figura 2.1.

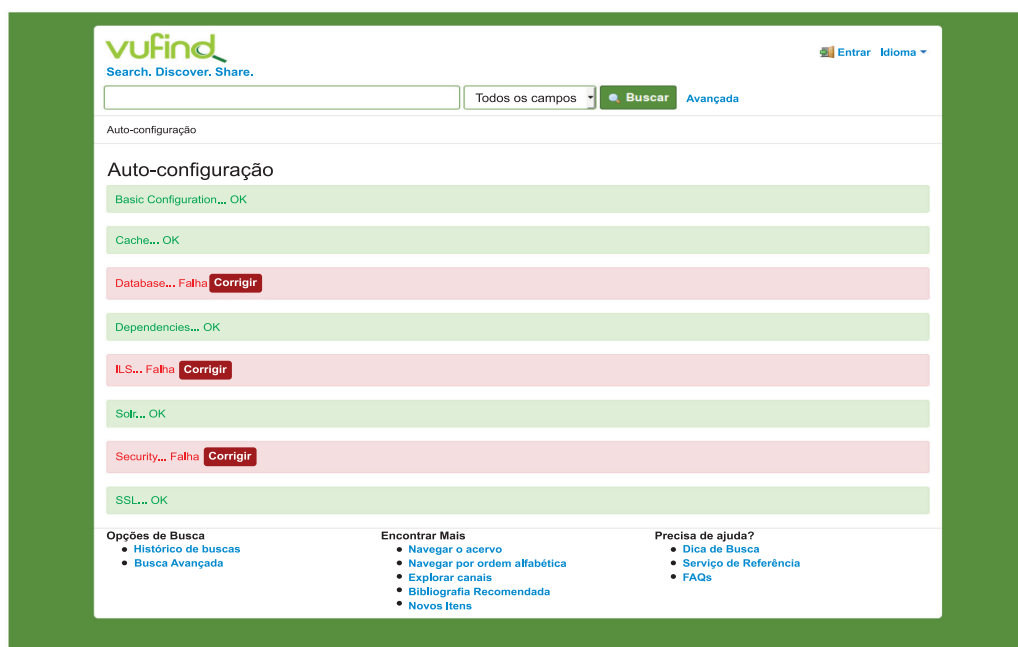


Figura 2.1 - Página de configuração no VuFind.

Fonte: Elaboração dos autores (2019).

Marcados em vermelho, têm-se os serviços que ainda não estão aptos a serem utilizados e necessitam ser habilitados.

Ao clicar no botão corrigir referente ao *Database*, Figura 2.2, traz os campos que necessitam ser preenchidos. Inicialmente, precisa escolher o sistema gerenciador de banco de dados que, neste livro, por opção, foi o *MySQL*. Um detalhe importante é que tanto o usuário quanto o banco de dados não precisam ser criados num momento anterior. A criação de ambos será de responsabilidade do usuário root do *MySQL* que é informado nos 2 últimos campos. Observe se a

senha do root está correta pois, caso contrário, a criação do banco de dados irá falhar. A instalação está sendo realizada num ambiente local; dessa forma, no campo referente ao host está sendo passado apenas *localhost*; todavia, se o *MySQL* estivesse alocado num servidor remoto, deveria passar IP do servidor (sem o *http*). Uma vez que todos os campos estiverem preenchidos, clique em enviar, se a criação obtiver sucesso, tem-se algo como mostrado na Figura 2.3.

Clicando no botão corrigir em *ILS*, escolha o campo **NoILS**, conforme Figura 2.4, e clique em *Submit Query*. Neste ponto, ainda estará apresentando a mensagem “*fail*”; para resolver, navegue pelo terminal até **/var/www/html/VuFind/local/config/VuFind** e abra o arquivo **NoILS.ini** e altere a terceira linha deste modo:

```
mode = none
```

Salve o arquivo e atualize a página e a mensagem de falha será removida. Por fim, clique no botão resolver em *Security* e tem-se algo como mostrado na Figura 2.5 indicando que a aplicação está apta a ser utilizada.

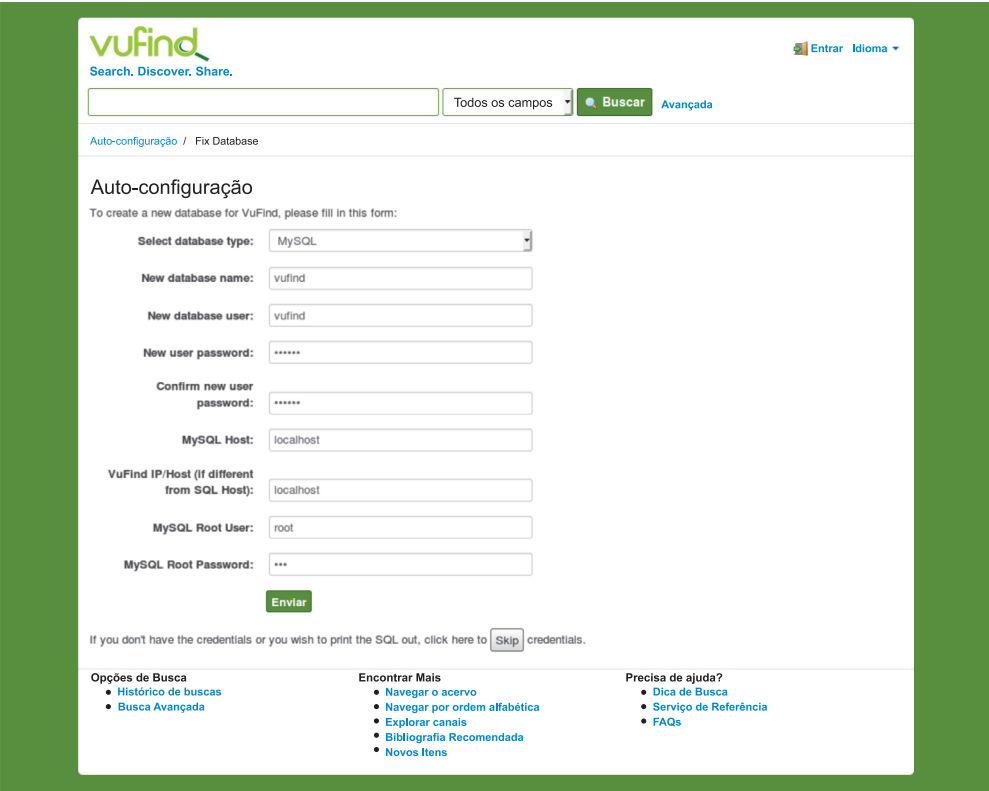


Figura 2.2 - Configuração do banco de dados no VuFind.
Fonte: Elaboração dos autores (2019).

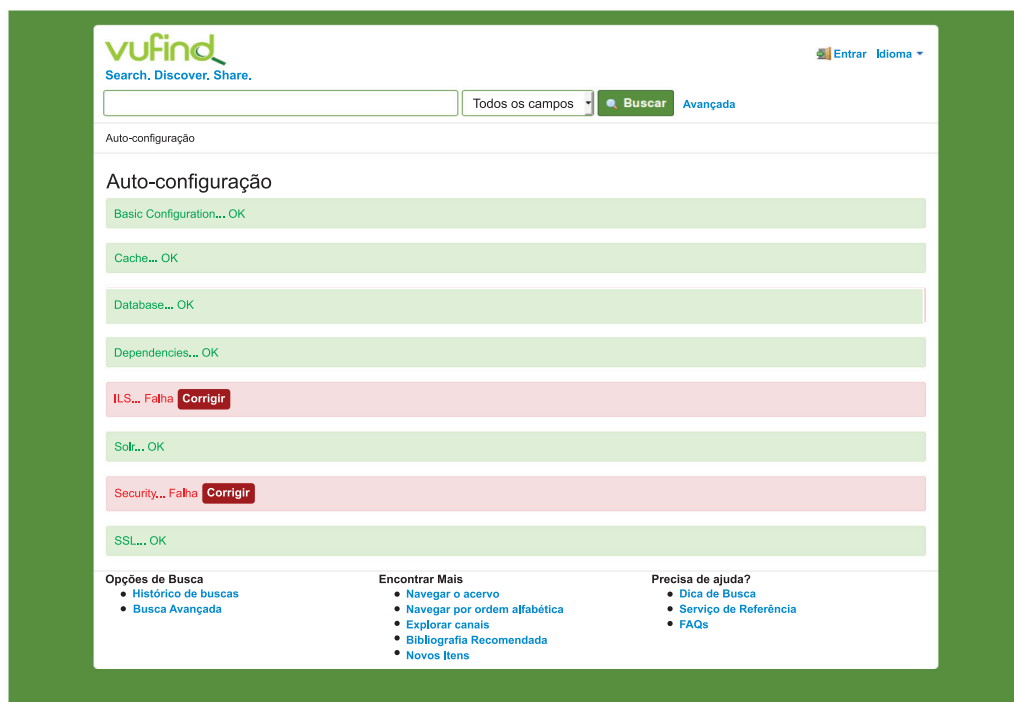


Figura 2.3 - Após criar o banco de dados, o aviso referente ao Database deve estar ok
Fonte: Elaboração dos autores (2019).

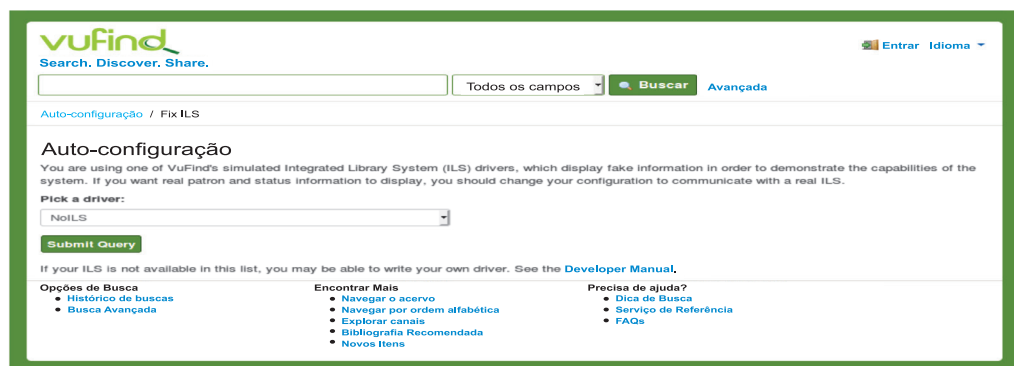


Figura 2.4 - Configuração do ILS.
Fonte: Elaboração dos autores (2019).

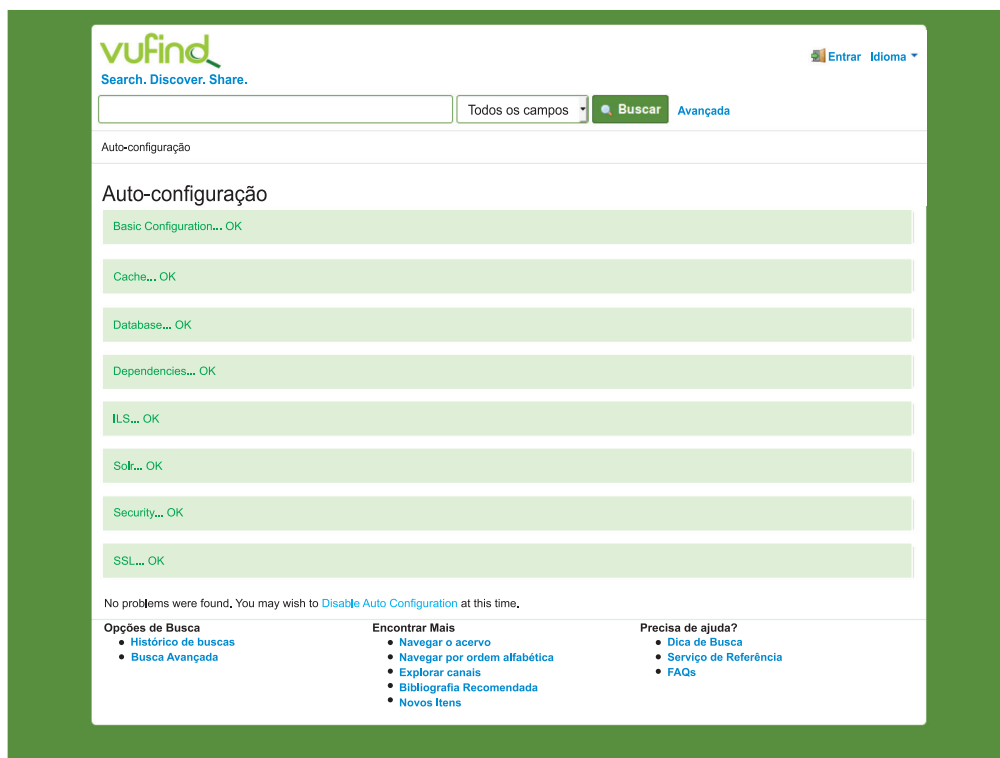


Figura 2.5 - VuFind configurado.

Fonte: Elaboração dos autores (2019).

2.5. Instalação de múltiplas instâncias

O VuFind possibilita a criação de múltiplas instâncias, em uma única instalação, e isso permite criar instâncias de acordo com as necessidades informacionais da instituição, facilitando a operação, o gerenciamento da aplicação e otimiza os recursos computacionais do servidor. Existem alguns motivos de a criação de múltiplas instâncias se fazer necessária, por exemplo:

- Necessidade de ter o mesmo índice de maneiras diferentes em URLs diferentes, ou seja, filtrar de maneira diferente o conteúdo da base ou oferecer temas diferentes para membros de uma rede ou consórcio;
- O VuFind ser usado como base para vários sistemas distintos, separados, rodando no mesmo servidor.

O VuFind foi desenvolvido para permitir que grande parte da configuração, incluindo a escolha do tema, seja definida por instância, enquanto permite que

essas opções inalteradas sejam herdadas da configuração principal. Existem algumas questões que devem ser analisadas, antecipadamente, ao se configurar uma nova instância:

- A nova instância usará o mesmo banco de dados de outras instâncias ou seu próprio banco? O banco de dados armazena as informações de usuários, favoritos e tags, então, a questão é se você deseja que a nova instância tenha os mesmos usuários e tags que as instâncias existentes, ou que tenha os seus próprios dados.
- A nova instância terá seu próprio núcleo Solr? O Núcleo Solr contém os registros bibliográficos pesquisáveis; se você quiser que a nova instância pesquise um conjunto separado de registros, ele precisará do seu próprio núcleo Solr.
- Seu sistema usará diretórios ou nomes de host para diferenciar sites diferentes?

A configuração da nova instância requer alguns passos que interferem, basicamente, em 3 partes da aplicação: configuração de uma nova instância, criação de um novo banco de dados e uma nova base no Solr.

2.5.1. Criando uma nova instância e configuração do diretório local

```
$ sudo php install.php --multisite
$ VuFind has been found in /usr/local/VuFind.
```

O *script* pede para informar onde serão armazenadas as configurações locais do novo módulo. Informe o nome do local:

```
$ Where would you like to store your local settings? [/usr/
local/vufind/local]<nomedolocal>
```

O *script* **install.php** suporta o uso de um módulo personalizado para armazenar alterações de código local. Se você decidir usar um módulo personalizado, o nome escolhido será usado para o nome do diretório do módulo e seu *namespace* PHP, caso contrário, pode pular esta etapa.

```
$ What module name would you like to use? [blank for none]
<nomedomodulo>
```

O *script* informa qual o nome do caminho que se deseja utilizar na nova instância. Coloque a barra “/” e nome do caminho. Vale ressaltar, por questões de

organização do sistema, que seja colocado o mesmo nome da instância, módulo e caminho, para facilitar a manutenção da aplicação.

```
$ What base path should be used in VuFind's URL? [/VuFind]
/<nomecaminho>
```

Em seguida, o sistema informa qual a configuração do virtual hosts do apache que deverá ser escolhida: 1) *Directory-based*, ou 2) *Host-based*. Em 1, por padrão, o VuFind é configurado para ser implantado em um diretório do seu site. No entanto, há a possibilidade de configurar o VuFind para carregar na raiz do seu domínio como segue:

1. Navegue até o **\$VuFind_HOME** e em **local/config/VuFind/config.ini** defina a configuração “url” como “http://seudominio.edu”. Posteriormente, edite o arquivo **VuFind/local/ httpd-VuFind.conf**, como segue:
 - a. Linha 2, altere “**AliasMatch^/VuFind2/themes/...**” para “**AliasMatch^/themes/...**”.
 - b. Remover a linha 34 referente a um “**Alias**” precedida pelo comentário “**Configuration for general VuFind base**”.
2. Por fim, será necessário acessar o arquivo em **/etc/apache2/sites-available/000-default.conf**. Caso esteja usando outro arquivo que não seja o *default* (**000-default.conf**), utilize-o para realizar as modificações a seguir:

```
NameVirtualHost *:80
<VirtualHost *:80>
    DocumentRoot /usr/local/VuFind/public
    ServerName seudominio.br
    ServerAlias www.seudominio.br
</VirtualHost>
```

Feche o arquivo e atualize o apache para reconhecer as alterações:

```
$ sudo service apache2 reload
```

O exemplo acima descreve os passos para mapear as instâncias do VuFind via *VirtualHost*. Todavia, neste livro, será abordada a segunda opção (*Directory-based*) onde é realizado o passo a passo para colocar 2 instâncias do VuFind em funcionamento, onde a segunda instância tanto pode herdar a base de dados da primeira quanto utilizar uma base de dados nova, o que na maioria das vezes é o desejado.

Segue a continuação da instalação, utilizando a opção *Directory-based*.
Observação: a opção “1”, abaixo, refere-se ao *Directory-based*.

```
$ Which option do you want? <opção> 1
```

No final, deve-se criar um link simbólico para o novo arquivo de configuração *httpd-VuFind.conf* dos sites no diretório Apache. Por exemplo, na versão 2.2 do Apache, execute o seguinte comando:

```
$ sudo ln -s /usr/local/VuFind/<nomenovainstancia>/httpd-VuFind.conf /etc/apache2/conf.d/VuFind<nomenovainstancia>.conf
```

No Apache versão 2.4, o diretório **conf.d** foi substituído pelo *conf-enabled* e *conf-available*. Segue o comando para criar o link simbólico na versão 2.4+ do apache:

```
$ sudo ln -s VUFIND_HOME/<nomenovainstancia>/httpd-VuFind.conf /etc/apache2/conf-enabled/VuFind<nomenovainstancia>.conf
```

2.5.2. Configurando a nova instância

As etapas a seguir devem funcionar para a maioria das configurações, porém, talvez seja necessário seguir as etapas sugeridas na instalação inicial na seção 2.2 - Instalação VuFind.

1. Reinicie o Apache para que a nova configuração tenha efeito.
2. Vá para **http://my-VuFind-server/<novainstancia>/Install/Home** e resolva todos os problemas:
3. Será informada a necessidade de executar comandos de permissão para o usuário **www-data**, conforme o comando abaixo:

```
$ chown -R www-data:www-data VUFIND_HOME/Vufind/<nomedanovainstancia>/config/VuFind
```

Se o diretório ‘Vufind’ não existir, será necessário criá-lo manualmente:

```
$ sudo mkdir VUFIND_HOME/Vufind/<nomedanovainstancia>/config/VuFind
```

Posteriormente, é necessário repetir o comando de permissões especificado em 2.

4. A interface da Web não permitirá que seja especificado o mesmo banco de dados existente. Caso deseje usar o mesmo banco de dados que outras

instâncias, será necessário especificá-lo no arquivo config.ini na seção '[Database]'. Navegue pelo terminal até:

```
$ VUFIND_HOME/Vufind/ <nomedanovainstancia>/config/VuFind/config.  
ini
```

2.5.3. Configurar um novo núcleo Solr

O Solr é responsável pela pesquisa dos registros através do VuFind. Caso queira criar um novo núcleo do Solr em sua nova instância, para que registros diferentes sejam indexados, devem ser seguidos os seguintes passos para finalizar a configuração do Solr no VuFind.

Os seguintes passos foram realizados na versão 5.0 do VuFind, podendo variar conforme versões anteriores.

- Parar a execução do Solr:

```
$ VUFIND_HOME/Solr.sh stop
```

- Copie o núcleo (core) existente, geralmente o núcleo “**biblio**”, localizado no diretório \$VUFIND_HOME/Solr/vufind para o novo diretório que você definiu:

```
$ cd VUFIND_HOME/Solr/vufind/  
$ cp -pr biblio <nome do novo núcleo>
```

- Se você não quer que o novo núcleo contenha registros do núcleo copiado, por exemplo do núcleo biblio, vá ao novo diretório e delete (ou mova por precaução) a pasta index:

```
$ cd VUFIND_HOME/Solr/vufind/<nome do novo núcleo>  
$ rm -r index
```

- Edite o arquivo \$VUFIND_HOME/Solr/vufind/<nome do novo núcleo>/conf/Solrconfig.xml para ajustar o caminho correto do novo núcleo. Localize o nó <dataDir> e faça o ajuste:

```
<dataDir>${Solr.Solr.home:./Solr}/<nome do novo núcleo></  
dataDir>
```

- Edite o arquivo config.ini da nova instância localizado em '\$VUFIND_HOME/<nome do novo núcleo>/config/VuFind/config.ini', e mude

o núcleo padrão abaixo da seção ‘**[index]**’ para o nome núcleo criado, por exemplo:

```
default_core = <nome do novo núcleo>
```

- Se for usar o SolrMarc (para importar os registros MARC21 para nova instância), você deverá configurar a diretiva **Solr.core.name**, com o nome do núcleo criado, localizado no arquivo **import/import.properties** no seu diretório local do seu novo núcleo.

```
$ VUFIND_HOME/<nome do novo núcleo>/import/import.properties  
Solr.core.name = <nome do novo núcleo>
```

Importante: Ao indexar, certifique-se de que a variável de ambiente **VUFIND_HOME** aponta para a configuração correta, para que os registros sejam enviados para o local apropriado. Essa variável precisa ser definida para que os cronkobs importem registros para diferentes núcleos.

2.5.4. Configure a nova instância

Na sequência, serão descritos detalhes de configuração básica, customizações e temas para uma nova instância.

2.5.4.1. Configuração básica

Se a nova instância estiver configurada quase exatamente como a instância anterior, mas com algumas pequenas modificações, considere usar a seção **[Parent_Config]** do config.ini para herdar configurações de uma configuração de base compartilhada. Isso facilitará a manutenção ao longo do tempo. A seção **[Parent_Config]** é suportada por todos os arquivos .ini do VuFind, logo, pode-se usar essa técnica em qualquer arquivo que precise ser personalizado.

2.5.4.2. Código personalizado

Caso seja necessário escrever código específico para uma determinada instância, ou personalizar o código de maneiras diferentes, para instâncias

diferentes, basta criar um módulo separado para cada instância. Isso pode ser realizado, executando-se o arquivo **install.php** no terminal.

```
$ php $VUFIND_HOME/install.php
```

Quando solicitar a criação de novo módulo, informar o nome.

```
VuFind supports use of a custom module for storing local code changes.
```

```
If you do not plan to customize the code, you can skip this step.
```

```
If you decide to use a custom module, the name you choose will be used for
```

```
the module's directory name and its PHP namespace.
```

```
What module name would you like to use? [blank for none]  
<moduloPersonalizado>
```

Acesse o diretório **VUFIND_HOME/Vufind/module/<modulo Personalizado>** do módulo de código personalizado para ver os detalhes para implementação do módulo recém-criado.

2.5.4.3 Tema personalizado

O tema do VuFind é determinado pelas configurações do config.ini na seção “Site” na diretiva “**theme**”. Por padrão, o Vufind disponibiliza três temas:

- bootstrap3 - tema *HTML5* usando bibliotecas *Bootstrap3* + *jQuery*, com estilo mínimo.
- bootprint3 - tema bootstrap3 com estilo padrão mais atraente aplicado.
- sandal - tema bootstrap3 com estilo mais limpo.

Como cada instância do VuFind tem seu próprio config.ini em seu diretório de configurações locais, você pode configurar facilmente temas separados para cada instância. Como os temas podem herdar um do outro, é possível criar um tema base para personalizações compartilhadas e isolar pequenas alterações distintas para temas muito pequenos específicos da instância. Consulte a documentação oficial¹ para mais detalhes.

¹ www.vufind.org.br

Capítulo 3

Indexação

Uma vez que o VuFind esteja implantando, pode-se iniciar a fase de inclusão de dados a partir de fontes que implementem o protocolo OAI-PMH. A seguir, será explicado o passo a passo para realizar a coleta dos metadados no VuFind utilizando OAI-PMH.

3.1. Harvesting OAI-PMH

O VuFind possui a funcionalidade de agregador de dados, onde a fonte de dados são aplicações que implementam o protocolo OAI-PMH. Na Figura 3.1, consta a arquitetura de *harvest* no VuFind, onde os *datasets* são fontes de dados disponibilizados em um ou mais formatos de descrição de recursos.

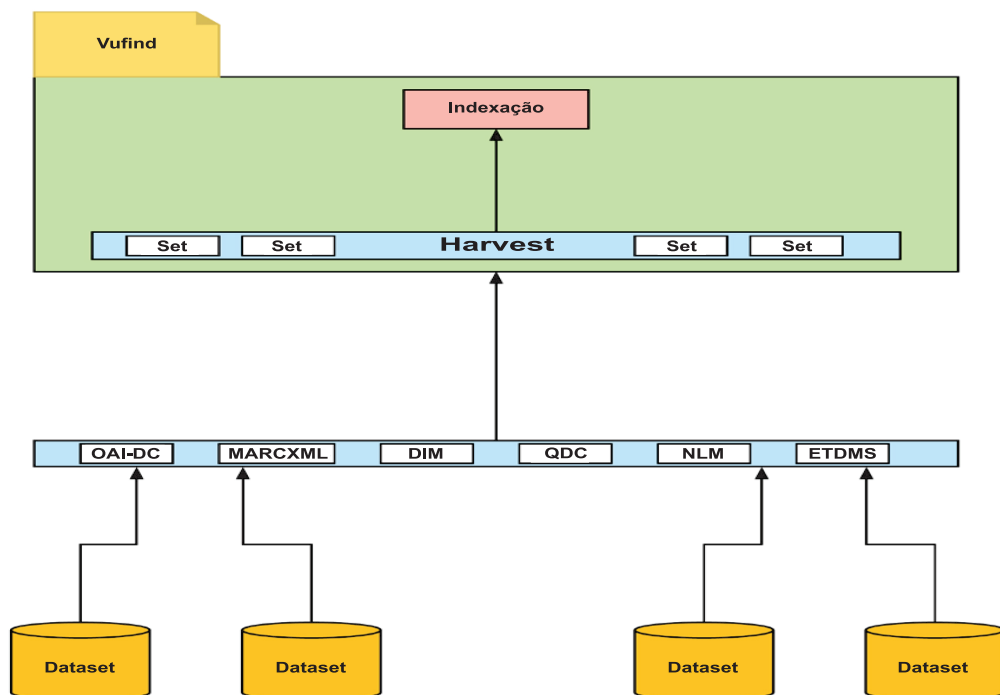


Figura 3.1 - Arquitetura de Harvest no VuFind.

Fonte: Elaboração dos autores (2019).

Neste trabalho, dá-se ênfase aos formatos mais utilizados:

- **OAI-DC**- provê mecanismo que facilita a recuperação de recursos na Internet através de 15 padrões de metadados: Title, Creator, Subject, Description, Published, Contributor, Date, Type, Format, Identifier, Source, Language, Relation, Coverage, Rights. O Quadro 3.1 apresenta um exemplo da utilização.

Elemento	Conteúdo
DC.Title	Dublin Core
DC.Creator	<i>Dublin Core Metadata Initiative (DCMI)</i>
DC.Subject	Metadados
DC.Description	Projeto da comunidade para desenvolver padrões de interoperabilidade de metadados.
DC.Published	DCMI
DC.Contributor	Comunidade mundial
DC.Date	2019
DC.Type	Recurso Interativo
DC.Format	text/html
DC.Identifier	http://www.ibict.br/ciconline/1234/exemplo.pdf
DC.Source	Internet
DC.Language	pt-BR
DC.Relation	Trabalho único
DC.Coverage	Brasil
DC.Rights	IBICT

Quadro 3.1 - Exemplo ilustrativo do formato OAI-DC.

Fonte: Elaboração dos autores (2019).

- **MARCXML** - viabiliza a comunicação de descrições bibliográficas em formato legível de computador. O formato é composto por 3 níveis:
 - 1) Estrutura física dos registros.
 - 2) Designação de conteúdo, onde os campos (tags), indicadores e delimitadores são responsáveis pela recuperação das informações.
 - 3) Conteúdos bibliográficos, funcionando como convenção de entrada das informações nos campos de informação: título, autor, editora, local, entre outros, normalmente definidos pelo ISBD² ou AACR2³.

Há uma grande quantidade de campos e subcampos que podem ser usados.

- 1) Campos variáveis que armazenam informações não estruturadas, de tamanho variável subdivididos em campos de controle e campos de dados. Os campos de controle não possuem código de controle, todavia, podem conter um elemento de informação ou uma série de dados com

² *International Standard Bibliographic Description (ISBD) padrão desenvolvido pela International Federation of Library Associations and Institutions para a criação de descrições bibliográficas.*

³ *Código de Catalogação Anglo-Americano (AACR2) compêndio de regras para a criação de descrições bibliográficas.*

tamanho fixo, identificados pela posição dos caracteres: Campos 001 (número de controle), 003 (identificador do número de controle), 005 (data e hora da última atualização do registro), 006 (material adicional), 007 (descrição física) e 008 (aspectos bibliográficos). Com relação aos campos de dados, tem-se a seguinte nomenclatura, nas faixas 01X até 8XX são utilizados dois tipos de designação de conteúdo, indicadores que podem ser um número de 0 a 9 ou # (quando o indicado não é utilizado), e os códigos de subcampos que, geralmente, são letras minúsculas com conteúdo a ser descrito. Para mais informações, consulte a página oficial do MARC21 da Library of Congress⁴.

- **DIM** - O esquema DIM é essencialmente uma maneira de representar a estrutura de metadados internos do Dspace⁵ em XML. Os metadados internos do Dspace são muito semelhantes a um *Qualified Dublin Core*, em sua estrutura, destinado principalmente a metadados descritivos. No entanto, os metadados do Dspace permitem que elementos sejam personalizados, qualificadores ou esquemas possam ser criados (é extensível a qualquer número de esquemas, elementos e qualificadores). Estes campos / esquemas personalizados podem ou não serem traduzidos para o *Qualified Dublin Core* normal. Dessa forma, o esquema DIM é capaz de expressar esquemas de metadados, elementos ou qualificadores que podem ou não existir no *Qualified Dublin Core*.
- **QDC** - Qualificador Dublin Core julgado pelo Comitê de Uso da DCMI⁶ em conformidade com os princípios de boas práticas para a qualificação dos elementos de metadados Dublin Core. A DCMI reconhece duas grandes classes de qualificadores:
- **Refinamento de elemento:** Esses qualificadores tornam o significado de um elemento mais restrito ou mais específico. Um elemento refinado compartilha o significado do elemento não qualificado, mas com um escopo mais restrito. Um cliente que não entende um termo de refinamento de elemento específico deve ser capaz de ignorar o qualificador e tratar o valor de metadados como se fosse um elemento não qualificado (mais amplo).

⁴ Disponível em: <<https://www.loc.gov/marc/>>

⁵ Software *open-source* que implementa uma biblioteca digital.

⁶ Dublin Core Metadata Initiative (DCMI) é uma organização dedicada a promover a adoção generalizada de padrões de metadados interoperáveis e desenvolver vocabulários de metadados especializados.

- **Esquema de codificação:** Esses qualificadores identificam esquemas que auxiliam na interpretação de um valor de elemento. Esses esquemas incluem vocabulários controlados e notações formais ou regras de análise. Um valor expresso usando um esquema de codificação será, portanto, um *token* selecionado de um vocabulário controlado (por exemplo, um termo de um sistema de classificação ou conjunto de cabeçalhos de assunto) ou uma *string* formatada de acordo com uma notação formal (“2000-01- 01 “como a expressão padrão de uma data) (Quadro 3.2).

Elemento	Refinamento de elemento	Esquema de codificação
Title	Alternative	-
Creator	-	-
Subject	-	LCSH MeSH DDC LCC UDC
Description	Table Of Contents Abstract	
Publisher	-	-
Contributor	-	-
Date	Created Valid Available Issued Modified	Período DCMI W3C-DTF
Type	-	Tipo de vocabulário DCMI
Format	Extent	-
Identifier	-	URI
Source	-	URI
Language	-	ISO 639-2 RFC 1766

Elemento	Refinamento de elemento	Esquema de codificação
Relation	Is Version Of Has Version Is Replaced By Replaces Is Required By Requires Is Part Of Has Part Is Referenced By References Is Format Of Has Format	URi
Coverage	Spatial	Ponto DCMI ISO 3166 DCMI Box TGN
Coverage	Temporal	Período DCMI
Rights	-	-

Quadro 3.2 - Exemplo de qualificadores QDC.

Fonte: Elaboração dos autores (2019).

- **NLM-** usa o padrão de referências bibliográficas ANSI / NISO Z39.29-2005 (R2010) como base para o formato de citações MEDLINE / PubMed para artigos de periódicos.
- **ETDMS-** Padrão de Metadados de Interoperabilidade para Teses e Dissertações Eletrônicas. Segue-se uma descrição dos elementos comuns de metadados *Dublin Core* (e um novo elemento especificamente para teses). Diretrizes são dadas a respeito de quais informações relacionadas a um ETD pertencem a cada elemento.

Elemento	Nota de descrição
dc.tile	Um nome dado ao recurso. No caso de teses e dissertações, este é o título do trabalho tal como aparece na página de rosto
dc.title.alternative	título alternativo da tese ou dissertação
dc.creator	responsável por criar o conteúdo do recurso
dc.subject	O tópico do conteúdo do trabalho
dc.description.abstract	Resumo do trabalho
dc.description.note	Adicionais informações sobre o trabalho

Quadro 3.3 - Campos ETDMS

Fonte: Elaboração dos autores (2019).

3.1.1. Coleta no VuFind

Anteriormente foi dada uma breve explicação de alguns populares formatos de dados aceitos pelo protocolo OAI-PMH. Daqui por diante, serão relatados os passos para coletar os *datasets* das aplicações que respondem ao protocolo OAI-PMH para o VuFind. Inicialmente, é necessário realizar algumas configurações no VuFind para designar as fontes de dados a serem coletadas no arquivo **oai.ini** localizado em **\$VUFIND_HOME/harvest/oai.ini**. A seguir, serão apresentados todos os parâmetros aceitos no VuFind para a implementação do *harvest*:

```
[NomeSection]
url = http://urlAplicação.br
set = meu_set_opcional
metadataPrefix = formato
timeout = 60
combineRecords = false
combineRecordsTag = <collection>
idSearch[] = "/^oai:nomedoHostdoSeuEspaço:/"
idReplace[] = "ir-"
idSearch[] = "\\/\\/\"
idReplace[] = \"-\"
injectDate = "datestamp"
injectId = "identifier"
injectSetName = false
injectSetSpec = false
injectHeaderElements[] = hierarchy
dateGranularity = auto
harvestedIdLog = harvest.log
verbose = false
autossllca = true
sslverifypeer = true
sanitize = true
sanitizeRegex[] = "[^\x{0009}\x{000a}\x{000d}\x{0020}-\x{D7FF}\x{E000}-\x{FFFF}]+/u"
badXMLLog = bad.log
httpUser = myUsername
httpPass = myPassword
```

Uma descrição de cada parâmetro é dada a seguir:

- “*NomeSection*” é o nome do diretório no qual os registros coletados são armazenados no VuFind (um subdiretório de *harvest* em **\$VUFIND_HOME**) e posteriormente será usado para coletar os dados.
- “*url*” é a URL base para a aplicação dos dados que se deseja coletar. Vale ressaltar que essa URL irá apontar para um link que responde ao protocolo OAI-PMH.
- “*set*” é o identificador de para coleta (normalmente encontrado na tag **<setSpec>** de uma resposta OAI-PMH **ListSets**). Você pode coletar vários conjuntos, colocando várias linhas “*set [] = x*” em sua configuração. Caso a aplicação tenha um único “*set*”, a tag pode ser omitida.
- “*metadataPrefix*” é o formato de metadados para coleta (**oai_dc** será usado por padrão; se o valor for omitido).
- “*timeout*” pode ser configurado para o número de segundos para uma solicitação OAI. Por padrão, o parâmetro é definido como 60.
- “*combineRecords*” pode ser configurado como *true* se você desejar que os registros coletados sejam combinados em número menor de arquivos (isso é determinado pelo tamanho do bloco de resposta do servidor OAI). A configuração padrão (*false*) resultará em um novo arquivo, sendo criado para cada registro. Essa função é destinada principalmente para a coleta de registros MARC21.
- “*combineRecordsTag*” pode ser usado para fornecer uma tag XML inicial e final (se o parâmetro *combinedRecords* estiver definido como *true*), que será usada para agrupar o conjunto de registros combinados.
- “*idSearch*” e “*idReplace*” pode ser usado para manipular IDs com expressões regulares. Isso é útil para adicionar ou remover prefixos e trocar caracteres problemáticos. Podem ser usados vários pares de pesquisa e substituir configurações para executar várias manipulações.
- “*injectId*” pode ser configurado para um nome de tag XML para injetar o ID do registro nos metadados coletados (contidos na tag especificada). Caso seja omitido ou definido como *false*, nenhuma alteração relacionada ao ID será feita nos metadados coletados.
- “*injectSetName*” pode ser configurado para um nome de tag XML para injetar o valor **setName** do registro nos metadados coletados (contidos

- na tag especificada). Se omitido ou definido como *false*, nenhuma alteração relacionada ao **setName** será feita nos metadados coletados.
- “*injectSetSpec*” pode ser configurado para um nome de tag XML para injetar o valor **setSpec** do registro nos metadados coletados (contidos na tag especificada). Se omitido ou definido como *false*, nenhuma alteração relacionada ao **setSpec** será feita nos metadados coletados.
 - “*injectHeaderElements*” pode ser definido para um *array* de elementos dentro do cabeçalho da resposta OAI-PMH que deve ser copiada para o documento XML salvo. Observação: isso raramente é necessário.
 - “*dateGranularity*” é a granularidade usada pelo servidor para representar datas. Isso pode ser “AAAA-MM-DDThh:mm:ssZ”, “AAAA-MM-DD” ou “auto” (consulta o servidor para obter detalhes). Por padrão, o parâmetro é configurado como “auto”.
 - “*harvestedIdLog*” é um nome de arquivo (dentro de seu diretório de coleta) para um arquivo de texto que lista todos os registros colhidos não excluídos. Se tal configuração for omitida, nenhum arquivo de log será gerado. Além disso, coletas subsequentes serão anexadas ao arquivo existente.
 - “*verbose*” pode ser configurado como *true* para exibir uma saída mais detalhada durante a coleta; isso pode ser útil para solucionar problemas. Por padrão, o parâmetro é definido como *false*.
 - “*autosslda*” tentará auto detectar sua autoridade de certificação SSL.
 - “*sslverifypeer*” pode ser configurado como *false* para desabilitar a verificação do certificado SSL; o padrão é *true* e a alteração da configuração não é recomendada.
 - “*sanitize*” pode ser configurado como *true* para remover caracteres ilegais de respostas XML. O padrão é *false*, assumindo que o servidor OAI-PMH ao qual a coleta está sendo feita fornecerá dados válidos.
 - *sanitizeRegex* pode ser definido como um *array* de strings regex, usado para limpar o XML recuperado de uma fonte OAI-PMH. Qualquer sequência de texto correspondente a essas expressões será substituída por espaços em branco.
 - “*badXMLLog*” pode ser definido para um nome de arquivo (que será criado dentro de seu diretório de coleta) para conter cópias de XML inválido que foi corrigido quando a configuração *sanitize* (acima) é definida como *true*.

- “*httpUser*” é um nome de usuário opcional, caso o repositório OAI esteja por trás da autenticação básica HTTP. Deve ser definido em combinação com o *httpPass*.
- “*httpPass*” é uma senha opcional no caso de o repositório OAI estar por trás da autenticação básica HTTP. Deve ser definido em combinação com o *httpUser*.
- A partir dos parâmetros listados acima, o usuário pode configurar o harvest conforme a necessidade. Lembramos que no arquivo *oai.ini* existe um exemplo contendo o mínimo de parâmetros a serem utilizados para se realizar um *harvest* no *Open Journal System* (OJS) onde é necessário apenas configurar o parâmetro URL, fornecendo um endereço de uma revista OJS.

Exemplo incluso no **oai.ini**:

```
[OJS]
url = http://ojs.myuniversity.edu/oai
metadataPrefix = oai_dc
idSearch[] = "/^oai:myuniversity.edu:/"
idReplace[] = "ojs-"
idSearch[] = "/\\//"
idReplace[] = "-"
injectId = "identifier"
injectDate = "datestamp"
```

Após configurar uma fonte provedora de dados no arquivo *oai.ini*, deve-se realizar a coleta dos dados em **VUFIND_HOME/harvest/** executando o seguinte comando.

```
$ php harvest_oai.php NomeSection
```

OBSERVAÇÃO

O comando acima vai coletar para um diretório específico, por exemplo no arquivo *oai.ini* tem um exemplo para coletar no OJS. Assim, o comando seria:

```
$ php harvest_oai.php OJS
```

Se existem várias fontes de dados configuradas no **oai.ini** execute o comando *default* para coletar de todas essas fontes:

```
$ php harvest_oai.php
```

3.2. Importação no VuFind

Após a coleta dos dados, será necessário indexar dados no sistema, para que seus usuários tenham algo que possam pesquisar. O indexamento será feito após o carregamento dos dados no Solr⁷. Há duas formas de realizar o *import*, uma onde os dados foram carregados conforme explicado na Seção 3.1.1 e uma onde os dados são exportados diretamente da aplicação desejada.

Inicialmente, serão relatados os campos possíveis de serem utilizados no Solr, sem a necessidade de realizar customizações. Posteriormente, como importar dados exportados de uma aplicação específica e, por fim, será descrito como importar os dados coletados (conforme Seção 3.1.1) diretamente no VuFind.

3.2.1. Campos Acessíveis no Solr

Nome do campo	Tipo	Descrição
version	long	usado internamente no Solr
allfields	text	engloba todos os textos pesquisáveis do registro, útil para juntar todos os campos e pesquisar a partir desse.
allfields_unstemmed	textProper	campo clonado do <i>all fields</i> , usado para ranking de relevância.
author	textProper	autores principais do trabalho (valor único até o VuFind 3.0, quando se tornou valor multivalorado)
author_additional	textProper	autores do índice; usado para buscas, mas não são exibidos.
author_browse	string	valor copiado dos campos <i>author</i> , <i>author2</i> e <i>author_corporate</i> ; usado para cabeçalho alfabético
author_corporate	textProper	autores corporativos da publicação (introduzido no VuFind 3.0)

⁷ Plataforma de pesquisa de código aberto, escrita em Java, do projeto Apache Lucene. Seus principais recursos incluem pesquisa de texto completo, indexação em tempo real, *clustering* dinâmico, integração de banco de dados como NoSQL e gerenciamento de documentos ricos

Nome do campo	Tipo	Descrição
author_corporate_role	string	os papéis dos principais autores corporativos da publicação. Ver o campo <i>author_corporate_role</i> (introduzido no VuFind 3.0)
author_facet	textFacet	nomes de autores para uso em facetamento; campo copiado de <i>author</i> , <i>author2</i> e <i>author_corporate</i> (introduzido no VuFind 3.0 para substituir <i>authorStr</i>)
author_fuller	textProper	forma mais completa do nome do (s) autor (es) principal (ais); usado para pesquisa e relevância, mas não exibido. Isso se tornou multivalorizado no VuFind 3.0, era valor único antes disso.
author_role	string	os papéis dos principais autores da publicação, geralmente indexados como códigos de relator que são traduzidos no tempo de exibição. Opcional, mas, se preenchido, os valores devem compartilhar a mesma ordem que o elemento de autor. (introduzido no VuFind 3.0)
author_sort	string	campo a ser usado para ordenação do <i>author</i> (introduzido no VuFind 3.0 para substituir <i>authorStr</i>)
author_variant	text	contém formas variantes do (s) nome (s) encontrado (s) no campo <i>author</i> (introduzido no VuFind 3.0)
author2	textProper	autores secundários da publicação
author2_fuller	textProper	o mesmo que <i>author_fuller</i> , mas para o campo <i>author2</i> (introduzido no VuFind 3.0)
author2_role	string	o mesmo que <i>author_role</i> , mas para o campo <i>author2</i> (introduzido no VuFind 3.0; substitui o <i>author2-role</i> não utilizado anteriormente)
author2_variant	text	o mesmo que <i>author_variant</i> , mas para o campo <i>author2</i> (introduzido no VuFind 3.0)
building	string	o <i>building</i> contendo o item representado pelo registro (usado no facetamento)

Nome do campo	Tipo	Descrição
callnumber-first	string	Forma mais simples em termos humano do significado do número de chamada de LC Amplo (mais amplo que o número de telefone); usado para funcionalidade de navegação e faceta.
callnumber-label	string	<i>The lead portion of the LC call number (before the dot), normalized to uppercase; used for “more like this” and browse functionality⁸</i>
callnumber-raw	string	<i>All LC call numbers for the record (may also be used for other call number schemes with minor adaptation⁹</i>
callnumber-search	callnumberSearch	<i>copyFielded from callnumber-raw; specially processed for search purposes</i>
callnumber-sort	string	<i>an LC call number normalized for sorting</i>
callnumber-subject	string	<i>a human-readable form of the broad LC call number’s meaning (more specific than callnumber-first)¹⁰</i>
collection	string	nome da coleção do registro indexado (usado para facetamento)
container_issue	text	número da edição do item que contém o registro
container_reference	text	referência de texto livre à aparência deste item dentro do contêiner (por exemplo, MARC21 campo 773\$g).
container_start_page	text	página inicial da aparência deste registro dentro do contêiner.
container_title	text	título do item que contém este registro (por exemplo, título do periódico para um registro em nível de artigo)
container_volume	text	número de volume do item que contém este registro.
contents	text	tabela de detalhes do conteúdo.
ctrlnum	string	número de controle do sistema (isto é, MARC21 035a); atualmente não utilizado.

⁸ A parte principal do número de chamado da LC (antes do ponto), em caixa alta; usado no recurso “mais como estes” e funcionalidades de pesquisa

⁹ O número de chamada completo da LC (pode ser utilizado para outros esquemas de classificação, contudo deve ocorrer pequenas modificações)

¹⁰ Número de chamada da LC legível por máquinas (mais específico do que callnumber-first)

Nome do campo	Tipo	Descrição
dateSpan	string	data de cobertura para um registro que se estende por um período de tempo (ou seja, um periódico - ver campo 362\$a);
description	text	resumo descritivo do item (usado para exibição)
dewey-full	string	Números de chamada Dewey normalizados para compatibilidade com a busca do número de chamada do VuFind.
dewey-hundreds	string	número Dewey na precisão de nível de centenas (usado para facetamento)
dewey-ones	string	O número de Dewey na precisão de um nível (usado para facetamento)
dewey-raw	string	A <i>string</i> Dewey não processada (para uso em Alphabetical Heading Browse).
dewey-search	callnumberSearch	campo copiado de dewey-raw; especialmente processado para fins de pesquisa.
dewey-sort	string	um único número de Dewey normalizado para uso na classificação
dewey-tens	string	número de Dewey com precisão de dez níveis (usado para facetamento)
edition	string	descrição da edição (valor único); usado para exibição
era	text	títulos de assunto era; usado para pesquisa (e para exibição em registros não-MARC21).
era_facet	textFacet	usado para facetamento (em regras padrão de MARC21, contém os subcampos d e y de muitos campos de cabeçalho de assunto)
first_indexed	date	
format	string	descreve o formato (livro, DVD, etc.) do registro para exibição e facetamento; o valor é copiado para <i>allfields</i> e <i>allfields_unstemmed</i> para pesquisa.
fullrecord	string	contém metadados brutos (geralmente MARC21) para uso dentro do VuFind; não se destina a ser pesquisável.

Nome do campo	Tipo	Descrição
fulltext	text	contém texto completo associado ao registro; usado para busca
fulltext_unstemmed	textProper	campo copiado do texto completo; usado para classificação de relevância.
genre	text	cabeçalhos de assuntos de gênero; usado para pesquisa (e para exibição em registros não-MARC21).
genre_facet	textFacet	usado para facetamento (em regras padrão de MARC21, contém subcampo v da maioria dos campos de cabeçalho de assunto).
geographic	text	cabeçalhos de assuntos geográficos; usado para pesquisa (e para exibição em registros não-MARC21).
geographic_facet	textFacet	usado para facetamento (em regras padrão de MARC21, contém subcampo z da maioria dos campos de cabeçalho de assunto)
hierarchy_browse	string	
hierarchy_parent_id	string	
hierarchy_parent_title	string	
hierarchy_sequence	string	
hierarchy_top_id	string	
hierarchy_top_title	string	
hierarchytype	string	
id	string	identificador único de cada registro
illustrated	string	aceita valores “ <i>Illustrated</i> ” ou “ <i>Not Illustrated</i> ” (usado para filtragem na pesquisa avançada)
institution	string	nome da instituição que pertence o registro (usado no facetamento)
is_hierarchy_id	string	
is_hierarchy_title	string	
isbn	isn	ISBN(s) do registro
issn	isn	ISSN(s) do registro
language	string	idioma(s) do registro
last_indexed	date	

Nome do campo	Tipo	Descrição
long_lat	geo	Coordenadas usadas para funcionalidade geográficas
long_lat_display	text	Latitude e longitude legível por humanos.
long_lat_label	string	Uma etiqueta descritivo associada com a latitude e longitude
lccn	string	LCCN (valor único) para o registro
marc_error	string	usado pelo SolrMarc para armazenar erros durante a indexação
oclc_num	string	número OCLC do registro
physical	string	descrição física do item (usado para visualização)
publishDate	string	Ano(s) da publicação
publishDateSort	string	Valor único selecionado de <i>publishDate</i> para uso em conjuntos de classificação
publisher	textProper	Nome do(s) editore(s)
publisherStr	string	Campo copiado do ' <i>publisher</i> ' não usado por padrão, mas pode ser usado para o facetamento
record_format	string	usado para controlar qual ' <i>record driver</i> ' é carregar para renderizar esse registro
series	text	Título da série, usado para busca e visualização
series2	text	título secundário da série usado para a busca, mas só mostrado quando a série está vazia.
spelling	textSpell	usado para gerar um dicionário de básico <i>spelling</i> ; campo copiado de <i>allfields</i>
spellingShingle	textSpellShingle	Usado para gerar um dicionário <i>shingled spelling</i> ; campo copiado de <i>author</i> , <i>title</i> , <i>contents</i> , <i>series</i> , <i>topic</i>
thumbnail	string	URL para a miniatura da imagem do item
title	text	título do trabalho (incluindo o subtítulo - e. g. 245a + 245b do MARC21)
title_alt	text	título(s) alternativo(s); usado para busca, mas não exibido atualmente
title_full	text	Declaração completa do título (incluído todos os subcampos alfabéticos do campo MARC21 245)

Nome do campo	Tipo	Descrição
title_fullStr	text	
title_full_unstemmed	textProper	
title_in_hierarchy	string	
title_new	text	título (s) mais novo (s) para o trabalho (geralmente usado para periódicos que mudam de nome)
title_old	text	título (s) mais antigo (s) para o trabalho (geralmente usado para periódicos que mudam de nome)
title_short	text	versão curta do título (e. g. 245a no MARC21)
title_sort	string	versão classificável do título
title_sub	text	subtítulo (e. g. 245b no MARC21)
topic	text	assuntos genéricos; usado para busca (e para exibição em registros não MARC21)
topic_browse	string	campo copiado de <i>topic</i> ; usado para 'Alphabetical Heading Browse'
topic_facet	textFacet	usado para facetamento (na regra padrão MARC21, contém subcampos 'a' e 'x' na maioria dos campos de assunto)
topic_unstemmed	topic_unstemmed	campo copiado de <i>topic</i> ; usado para classificação de relevância
url	string	URL para o texto completo do registro

Quadro 3.4 - Campos no Solr.

Fonte: Elaboração dos autores (2019).

3.2.2. Importação de Registro MARC21 de uma Aplicação Específica

O VuFind possui uma ferramenta denotada SolrMarc para importar os registros. Os passos para realizar a importação é dada como segue:

1. Exportando os registros da aplicação:

Cada aplicação possui um formato específico de expor e exportar seus registros em MARC21; dessa forma, tal discussão será deixada à parte para o usuário.

2. Configurando a importação:

A ferramenta de importação depende das configurações localizadas em **VUFIND_HOME/import/import.properties**. Tal arquivo, além de fornecer informações sobre o Solr, pode interessar ao usuário, visto que há a possibilidade de estar sendo utilizado mais de uma instância do VuFind. Posteriormente, é necessário configurar o arquivo **VUFIND_HOME/import/ marc.properties**. Neste arquivo, podem ser configurados detalhes de nome de coleção sendo os registros importados alocados no VuFind, bem como o nome da instituição e o nome da biblioteca, além de outros parâmetros que podem ser de interesse.

3. Importando os registros:

Supondo que os registros a serem importados tenham sido tomados no passo 1, necessita-se executar o comando abaixo, utilizando o caminho para o “diretório” onde se encontram os registros:

```
$ VUFIND_HOME/import-marc.sh /diretório/registros.mrc
```

4. Reiniciando o Solr:

```
$ VUFIND_HOME/Solr.sh restart
```

5. Otimizando os índices:

Para melhorar o desempenho, é recomendável otimizar o índice do Solr após importar os registros. O comando abaixo é o *default* e irá otimizar unicamente o *biblio* no Solr.

```
$ VUFIND_HOME/util/optimize.php
```

Caso esteja sendo utilizada mais de uma instância do VuFind, é necessário incluir o nome da instância no Solr como argumento:

```
$ VUFIND_HOME/util/optimize.php instância_Solr
```

3.2.3. Importação de Registros XML Utilizando Transformações

Se os dados que você deseja importar não estiverem disponíveis no formato MARC21, é possível acessá-los em algum tipo de XML. Felizmente, carregar o XML no índice do VuFind é simples, se o informático for familiarizado com a linguagem **xslt**, pois basta traduzir do formato XML disponível para o formato de mensagem XML do Solr e, em seguida, enviar o resultado para o servidor do Solr. A ferramenta **xslt** do VuFind foi projetada para simplificar a publicação de

documentos transformados em **xslt** no índice Solr, oferecendo flexibilidade para estender o **xslt** e aplicar personalizações locais.

A ferramenta **xslt** é guiada por um arquivo de propriedades que fornece algumas informações importantes:

- O nome do arquivo **xslt** a ser usado.
- Os nomes das funções e classes customizadas do PHP que serão chamadas no arquivo **xslt**.
- Valores personalizados que se deseja passar como parâmetros para o arquivo **xslt** (exemplo: nomes de instituições locais, prefixos de ID, etc.)

Observe este exemplo de um arquivo de propriedades, onde linhas com “;” são relacionadas a comentários e as demais linhas são comandos específicos definidos como argumentos no arquivo.

; Configurações de importação XSLT

[General]

; NECESSÁRIO: Nome do arquivo XSLT a ser aplicado. O caminho é relativo ao diretório

; import/xsl; da instalação do VuFind.

xslt = exemplo.xsl

; OPCIONAL: PHP função (s) para registrar para uso no arquivo XSLT. Você pode repetir esta linha para registrar múltiplas funções do PHP.

;php_function[] = str_replace

; OPCIONAL: classe PHP preenchida com funções estáticas públicas para uso no arquivo XSLT. A classe deve estar dentro de um namespace PHP. Você pode especificar um nome totalmente qualificado

; se você não incluir um namespace, a classe será

; automaticamente considerada como residindo no namespace /VuFind/XSLT/Import.

custom_class[] = VuFind

;OPCIONAL: Se true, todas as configurações custom_class serão passadas para o

;XSLT com seus namespaces removidos; por exemplo, /VuFind/XSLT/Import/VuFind

;seria tratado como /VuFind em arquivos XSLT. Isso permite uma sintaxe mais compacta

;nos arquivos XSLT, mas pode levar a conflitos de nome se usado de maneira descuidada. Se definido como falso, deve-se usar nomes totalmente qualificados em seu; XSLT. A configuração false é recomendada, mas o padrão atual é "true" para

; compatibilidade com configurações herdadas.

truncate_custom_class = true

;Parâmetros XSLT - qualquer par chave/valor definido aqui será passado como parâmetros para o arquivo XSLT, permitindo que valores locais sejam definidos sem modificar o código XSLT.

[Parameters]

institution = "Minha Instituição"

collection = "Minha Coleção"

Após configurar o arquivo de propriedades, cujo nome fictício é **minhasConfigs.properties**, é necessário criar e customizar o arquivo **exemplo.xml** citado em **minhasConfigs.properties**, será em tal arquivo que será implementado e customizado os *beans* (contém código para refinar os dados internos em cada registro) que serão lidos os registros a serem importados.

- 1 - Crie o arquivo **exemplo.xml** em **VUFIND_HOME/import/xml/**
- 2 - Campos default a serem utilizados no arquivo **exemplo.xml**:

```
<!-- available fields are defined in solr/biblio/conf/schema.xml -->
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc/"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:php="http://php.net/xsl"
  xmlns:xlink="http://www.w3.org/2001/XMLSchema-instance">
  <xsl:output method="xml" indent="yes" encoding="utf-8"/>
  <xsl:param name="institution">Instituição</xsl:param>
  <xsl:param name="collection">Coleção</xsl:param>
  <xsl:param name="urlPrefix">http</xsl:param>
  <xsl:template match="oai_dc:dc">
    <add>
      <doc>

<!-- Campo injetado pelo harvester OAI-PMH -->
<field name="id">
<xsl:value-of select="//identifier"/>
</field>

<!-- RECORDTYPE -->
<field name="recordtype">dspace</field>

<!-- Instituição -->
<field name="institution">
<xsl:value-of select="$institution" />
</field>

<!-- Coleção -->
<field name="collection">
<xsl:value-of select="$collection" />
</field>

<!-- Idioma-->
<xsl:if test="//dc:language">
<xsl:for-each select="//dc:language">
<xsl:if test="string-length() > 0">
<field name="language">
```

```
<xsl:value-of select="php:function('VuFind::mapString', normalize-
space(string(.)), 'language_map_iso639-1.properties')"/>
</field>
</xsl:if>
</xsl:for-each>
</xsl:if>

<!-- Assunto -->
<xsl:if test="//dc:subject">
<xsl:for-each select="//dc:subject">
<xsl:if test="string-length() > 0">
<field name="topic">
<xsl:value-of select="normalize-space()"/>
</field>
</xsl:if>
</xsl:for-each>
</xsl:if>

<!-- Tipo -->
<xsl:if test="//dc:type">
<field name="format">
<xsl:value-of select="//dc:type" />
</field>
</xsl:if>

<!-- Autor -->
<xsl:if test="//dc:creator">
<xsl:for-each select="//dc:creator">
    <xsl:if test="normalize-space() ">
    <field name="author">
    <xsl:value-of select="normalize-space()"/>
    </field>
    <!-- use first author value for sorting -->
<xsl:if test="position()=1">
    <field name="author_sort">
    <xsl:value-of select="normalize-space()"/>
</field>
</xsl:if>
    </xsl:if>
</xsl:for-each>
</xsl:if>
```

```
<!-- Título -->
<xsl:if test="//dc:title[normalize-space()]">
  <field name="title">
    <xsl:value-of select="//dc:title[normalize-space()]" />
  </field>
  <field name="title_short">
    <xsl:value-of select="//dc:title[normalize-space()]" />
  </field>
  <field name="title_full">
    <xsl:value-of select="//dc:title[normalize-space()]" />
  </field>
  <field name="title_sort">
    <xsl:value-of select="php:function('VuFind::stripArticles', string(//
dc:title[normalize-space()])))" />
  </field>
</xsl:if>

<!-- Data -->
<xsl:if test="//dc:date">
  <field name="publishDate">
    <xsl:value-of select="substring(//dc:date, 1, 4)" />
  </field>
  <field name="publishDateSort">
    <xsl:value-of select="substring(//dc:date, 1, 4)" />
  </field>
</xsl:if>

<!-- URL -->
<xsl:for-each select="//dc:identifier">
  <xsl:if test="substring(., 1, string-length($urlPrefix)) = $urlPrefix">
    <field name="url">
      <xsl:value-of select="." />
    </field>
  </xsl:if>
</xsl:for-each>
</doc>
</add>
</xsl:template>
</xsl:stylesheet>
```

Inicialmente, é válido dizer que os campos que estão sendo descritos no exemplo.xsl estão pegando dados no formato *Dublin Core* (dc). Além disso, os campos descritos na `<field name="*">` por padrão são reconhecidos no Solr, ou seja, não há necessidade de configurá-los para utilizá-los. Todo conteúdo que está entre “<!--” e “-->” em **xslt** é considerado comentário. Pode-se ver que em alguns campos são utilizadas funções PHP (chamadas através do **php:function(...)**) para tratar o fluxo de dados. Tais funções podem ser tanto nativas no php, ou criadas pelo desenvolvedor, de acordo com as necessidades de customizações.

Para verificar se as transformações estão de acordo com o especificado, pode-se executar o comando abaixo:

```
$ cd VUFIND_HOME/import
$ php import-xsl.php --test-only XML_file properties_file
```

- **--test-only** - *flag* de teste definida, que mostra no terminal o resultado das transformações aplicadas no arquivo XML. Esta *flag* é útil para verificar se há erros nas transformações.
- **XML_file** - arquivo xml para o teste.
- **properties_file** - arquivo de configuração que aponto para o arquivo **xslt**.

Uma vez realizado o teste de importação com sucesso, pode-se importar os registros coletados. O arquivo **.properties** define o arquivo **.xslt** utilizado para normalizar os registros coletados; alterne para o subdiretório de importação da instalação do VuFind a partir do comando abaixo:

```
$ VUFIND_HOME/harvest/batch-import-xsl.sh VUFIND_HOME/local/
harvest/ArquivosparaImportar minhasConfigs.properties
```

Durante a execução do script acima, aparecerá no terminal a mensagem de “*Successfully imported*” junto ao nome do arquivo xml que foi importado.

Capítulo 4

Adicionando Facetas

As seções posteriores apresentaram como criar facetas utilizando diferentes formatos de dados. Na Seção 4.1, é apresentado o conteúdo referente à criação de facetas baseando-se em registros no formato MARC21. Criação de facetas utilizando registros no formato **xml** são apresentadas na Seção 4.2.

4.1. Facetamento com Registros MARC21

As facetas dependem de um campo de índice onde é pressuposto um dos três cenários possíveis:

1. O campo de índice a ser usado já existe.
2. Nenhum campo de índice existe atualmente e os dados no registro MARC21 serão indexados diretamente (ou seja, os dados não necessitam ou não se deseja normalização).
3. Nenhum campo de índice existe atualmente; os dados são codificados e precisam ser traduzidos em *strings* de texto primeiro.

4.1.1. Existe o Campo de Índice

Primeiro, deve-se verificar a existência do campo de índice e se tal campo contém os dados desejados. O arquivo **VUFIND_HOME/import/marc.**

properties contém o mapeamento padrão dos campos MARC21 para o índice do Solr. Deve-se atentar ao fato de que cada linha do arquivo preenche um único campo de índice. O nome do campo de índice Solr é anterior ao sinal de igual. Imediatamente após o sinal de igual, uma ou mais combinações do número de tag MARC21 de três dígitos e quaisquer subcampos são indexados juntos. Um “:” separa diferentes campos ou subcampos MARC21 para serem indexados separadamente. Um número entre parênteses indica a posição de byte a ser indexada. Ao inserir “*first*” no final da linha significa que apenas o primeiro valor será indexado.

O arquivo **VUFIND_HOME/config/facets.ini** contém uma lista de facetas a serem visualizadas. Cada linha é uma faceta incorporada. O exemplo a seguir apresenta um campo fictício no Solr (*SolrIndexName*) com o rótulo a ser visualizado no VuFind:

```
SolrIndexName = "Nome da faceta a ser visualizada no VuFind"
```

Numa instalação limpa, isto inclui as duas primeiras facetas, Instituição e Biblioteca que podem não ser necessárias. Tais facetas podem ser comentadas por inserir um ponto e vírgula (“;”) no início da linha. Por exemplo, para adicionar “data de publicação” como uma faceta, basta adicionar o indexador do campo no Solr, adicionando a seguinte linha no arquivo (facets.ini):

```
publishDate = "Ano de Publicação"
```

Vale ressaltar que as facetas são exibidas em ordem; caso a faceta *publishDate* seja adicionada na parte inferior da lista, ela será exibida na parte inferior da caixa de pesquisa no VuFind. Além disso, caso o campo adicionado não exista ou o arquivo apresente erro de sintaxe, a caixa de pesquisa não será visualizada.

4.1.2. Campo de Índice Inexistente sem Necessidade de Normalização

Se a faceta desejada ainda não for um campo de índice existente no Solr, deve-se primeiro criar o campo. Isso exigirá a re-indexação de todos os registros no VuFind.

O arquivo **VUFIND_HOME/import/marc.properties** mapeia campos e subcampos MARC21 para um índice.

Por exemplo, suponha que seja necessário adicionar os dados nos subcampos a e b, referentes aos campos 600, 650 e 651 a um índice denotado “IndiceNovo”.

Para fazer isso, é necessário adicionar a seguinte linha ao **marc.properties**. As melhores práticas relacionadas à arquitetura do software sugerem que arquivos como **marc.properties** sejam armazenados em diretórios de configurações locais que isolam suas alterações personalizadas do código principal do VuFind.

Ao colocar as customizações em **VUFIND_HOME/local/import/marc.properties** em vez de **VUFIND_HOME/import/marc.properties**, as linhas em **marc.properties** substituirão as linhas equivalentes em **marc.properties**. Isso permite que customizações especifiquem suas alterações em um só lugar, facilitando a atualização de alterações e adições à configuração principal durante upgrades no futuro. Dessa forma, faz-se necessário copiar o arquivo **marc.properties** para o diretório local como segue:

```
$ cp VUFIND_HOME/import/marc.properties VUFIND_HOME/local/import/marc.properties
```

Em seguida, abra o arquivo **VUFIND_HOME/local/import/marc.properties** no terminal e acrescente a linha abaixo:

```
IndiceNovo=655ab:650v:600v:651v
```

Em seguida, é necessário configurar o novo índice no Solr. O arquivo **VUFIND_HOME/solr/vufind/biblio/conf/schema.xml** define os campos no Solr. Na seção <fields>, adiciona-se a seguinte a linha:

```
<field name="IndiceNovo" type="textFacet" indexed="true" stored="true" multiValued="true" termVectors="true"/>
```

Posteriormente, é necessário re-indexar os registros no Solr, para que o índice “IndiceNovo” seja reconhecido. A faceta pode ser adicionada ao arquivo **facets.ini** conforme descrito acima.

Para re-indexar os registros no Solr, é necessário navegar pelo terminal até o diretório **VUFIND_HOME/local/harvest/**, e, em seguida, acessar os diretórios contendo os registros coletados. Supondo que um dos diretórios coletados cujo nome fictício seja “DirColetado”, para re-indexar os seus registros, execute os comandos abaixo num terminal Linux:

```
$ cd DirColetado
$ cd processed
$ mv * ../
```

4.1.3. Campo de Índice Inexistente e Tradução Necessária

Para dados codificados (como dados encontrados nos campos 007, 008), deve-se primeiro mapear os dados para sequências de texto. Felizmente, o formato MARC21 está bem documentado e listas do que cada código significa estão prontamente disponíveis nas Listas de códigos MARC21 e na página Formatos e padrões da OCLC.

Como exemplo, será criado o arquivo **VUFIND_HOME/ import/ translation_maps/ instrument_map.properties** para conter o mapeamento. O arquivo converterá os códigos de duas letras usados no campo MARC21 048 em texto legível. Cada linha do arquivo contém um único código possível e sua tradução.

```
ka = Piano
kb = Organ
kc = Harpsichord
kd = Clavichord
```

Na sequência, faz-se necessário adicionar a seguinte linha em **VUFIND_HOME/local/import/marc.properties**:

```
instrument_facet = 048a[0-1], instrument_map.properties
```

OBSERVAÇÃO

Os números entre parênteses indicam que o VuFind deve considerar apenas os dois primeiros bytes no subcampo 048a. A vírgula separa as informações do campo do nome do arquivo usado para traduzir os dados, neste caso, **instrument_map.properties**.

Por fim, adicione o novo índice no Solr:

```
<field name="instrument_facet" type="textFacet" indexed="true"
stored="true" multiValued="true" termVectors="true"/>
```

A partir deste ponto, a faceta está apta a ser utilizada no VuFind.

4.2. Facetamento com Registros XML

A criação de facetas a partir de registros em **xml** é semelhante ao que foi discutido em relação a registros em MARC21; por razões de completude, será dada uma explicação sobre os procedimentos necessários.

Ao criar uma faceta, dois cenários são possíveis:

- 1 - O índice a ser utilizado já está configurado no Solr;
- 2 - O índice não existe no Solr.

4.2.1. *Índice Existe no Solr*

Se o índice já está configurado no Solr, o procedimento de facetamento é simples. Inicialmente, é necessário mapear o campo desejado no arquivo **.xslt** no índice a ser utilizado no Solr.

Um exemplo clássico é identificar a coleção que será usada para organizar os registros provenientes de uma determinada aplicação. Isto pode ser feito da seguinte forma.

- 1 - Seja **minhasConfigs.properties** o nome fictício dado ao arquivo onde as configurações das propriedades de importação serão realizadas. Para criar ou alterar o arquivo, é necessário navegar até **VAR_HOME/import/minhasConfigs.properties** e alterar unicamente os parâmetros **xslt** e **collection** com a informação desejada, como segue:

```
xlst= transforma.xml  
collection = "Coleção Vufind"
```

- 2 - Arquivos **xslt**, são utilizados para aplicar as transformações necessárias, bem como indexar os dados aos campos mapeados no Solr. Em 1 o arquivo **xlst** fornecido foi o “transforma.xml”, dessa forma é necessário criar ou abrir o arquivo localizado em **VAR_HOME/import/xslt/transforma.xml** e acrescentar o conteúdo abaixo:

```
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc/"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:php="http://php.net/xsl"
  xmlns:xlink="http://www.w3.org/2001/XMLSchema-instance">
  <xsl:output method="xml" indent="yes" encoding="utf-8"/>
  <xsl:param name="collection">Coleção Vufind</xsl:param>
  <xsl:param name="urlPrefix">http</xsl:param>
  <xsl:template match="oai_dc:dc">
    <add>
      <doc>
        <!-- COLLECTION -->
        <field name="collection">
          <xsl:value-of select="$collection" />
        </field>
      </doc>
    </add>
  </xsl:template>
</xsl:stylesheet>
```

OBSERVAÇÃO

Acima, não está sendo aplicada nenhuma normalização ou mesmo mapeamento de dados em índices do Solr, com exceção do índice "collection". Além disso, vale ressaltar que o conteúdo que está sendo mapeado no índice é estático, tomado da variável **\$collection**. Todavia, tal conteúdo poderia ser dinâmico e mapeado com dados fornecidos a partir de um arquivo **xml**. Outro ponto é que, utilizando qualquer índice, seria possível criar facetas sem maiores dificuldades.

4.2.2. Índice não Existe no Solr

Caso haja a necessidade de criar uma faceta cujo índice não exista no Quadro x, será necessário criar o índice, conforme abaixo:

```
field name="NomeFaceta" type="textFacet" indexed="true"
stored="true" multiValued="true" termVectors="true"/>
```

Posteriormente, basta executar os procedimentos detalhados em 4.2.1.

4.3. Navegando por Ordem Alfabética

O módulo *AlphaBrowse* fornece uma navegação por título alfabético, permitindo aos usuários ver todos os autores, assuntos e títulos em sua coleção, incluindo referências cruzadas extraídas de dados de autoridade. Para aproveitar essa funcionalidade, é necessário adicionar uma etapa extra ao processo de indexação no intuito de gerar a base de dados a ser consultada. Para construir tal base de dados usada no módulo *AlphaBrowse*, necessita-se executar o comando abaixo:

```
$ VUFIND_HOME/index-alphabetic-browse.sh
```

Vale ressaltar a necessidade de executar o comando acima, sempre que novos dados forem coletados ao VuFind, dessa forma, mantêm-se os índices de busca alfabética sincronizados com a base de dados da aplicação. Caso o VuFind apresente problemas após executar o script, é indicado reiniciar o Solr.

4.4. Alterando a Busca

Por padrão, a busca oferece 5 metadados para consulta:

- Autor
- Assunto
- Área/Cota
- ISBN/ISSN
- TAG

Porém, há a possibilidade tanto de retirar quando incluir novos metadados, sendo a única exigência que o novo metadado, caso haja necessidade de inclusão, esteja mapeado em um campo Solr (veja quais campos no Quadro 3.4 no Capítulo 3).

Instruções para inclusão de um novo campo na busca:

1. Copiar os arquivos `search.ini` e **`searchspecs.yaml`** para as configurações locais:
2. Incluir campo Solr que se deseja na busca no arquivo `searchspecs.yaml` na Seção “*These searches use Dismax when possible*”:

```
campo_Solr:
  DismaxFields:
    - campo_Solr
  DismaxHandler: edismax
```

```
$ cp VUFIND_HOME/config/vufind/searchbox.ini searchspecs.yaml
VUFIND_HOME/local/config/vufind/
```

3. Incluir campo na Seção *Advanced_searches* no arquivo `search.ini.yaml`:

```
campo_Solr = novoMetadado
```

OBSERVAÇÕES

No item 2, deve-se colocar o campo Solr que irá receber o metadado na importação, uma vez que será através desse campo que os dados são encontrados. No item 3, é onde se faz o mapeamento para que o novo campo se torne visível ao usuário para eventuais utilizações.

É simples realizar a exclusão de um metadado na busca, bastando retirar o mapeamento que é visível ao usuário no arquivo **`searchspecs.yaml`** na Seção *Advanced_searches*.

4.5. Hierarquias e Coleções

Algumas coleções de documentos são estruturadas hierarquicamente. O VuFind oferece um mecanismo para exibir essas hierarquias como um *pop-up* nos resultados da pesquisa e como uma guia da página de registro.

Para exibir hierarquias, o VuFind precisa ser capaz de encontrar informações sobre relacionamentos de documentos em seu índice no Solr. No momento da indexação, é necessário preencher os seguintes campos para usar a funcionalidade de hierarquia:

- *hierarchytype*- O tipo de hierarquia sendo usado (isso aciona o carregamento de um driver de hierarquia; geralmente é recomendado deixá-lo em branco, fazendo com que o VuFind seja o padrão especificado para o driver especificado na seção [Hierarquia] do config.ini).
- *hierarchy_browse*- campo opcional.
- *hierarchy_top_id*- O ID do Solr do registro que representa o topo da hierarquia que contém o registro atual. Isso pode ter vários valores, se um registro pertencer a várias coleções.
- *hierarchy_top_title*- O título do registro representando o topo da hierarquia contendo o registro atual.
- Pode ter vários valores, se um registro pertencer a várias coleções; a ordem de vários valores no campo deve corresponder aos valores em *hierarchy_top_id*.
- *hierarchy_parent_id*- O identificador Solr do registro imediatamente acima do registro corrente na hierarquia.
- *hierarchy_parent_title*- O título do registro imediatamente acima do registro atual na hierarquia.
- Pode possuir múltiplos valores, se o registro pertence a múltiplas coleções, tal que a ordem dos múltiplos valores no campo deve corresponder aos valores em *hierarchy_parent_id*.
- *is_hierarchy_id*- se o registro atual representar um nodo dentro de uma hierarquia, esse campo deverá conter o mesmo valor que o campo identificador regular.
- *hierarchy_sequence*- Um valor que indica a posição do registro atual relativo a seus irmãos (mesmo nível hierárquico), usado para ordenar registros ao renderizar árvores.
- Pode ter vários valores, se um registro pertencer a várias coleções. A ordem de vários valores em tal campo deve corresponder aos valores contidos em *hierarchy_parent_id*.

- *is_hierarchy_title*- se o registro atual representar um nodo dentro de uma hierarquia, esse campo deverá conter o mesmo valor que o campo de título regular.
- *title_in_hierarchy*- Opcional, introduzido no VuFind 2.3. Se um registro precisar ter um título de exibição diferente em contextos de coleções diferentes, tal campo deverá conter um número de títulos igual ao número de coleções pai e na mesma ordem do campo *hierarchy_parent_id*.

Após o índice do Solr ser preenchido adequadamente, é necessário ajustar vários arquivos de configuração do VuFind para tornar as hierarquias visíveis. Altere o arquivo **VUFIND_HOME/config/vufind/config.ini**¹¹.

Há, também, a possibilidade de usar o suporte de hierarquia para links simples entre contêineres e seus conteúdos. Geralmente, essa funcionalidade é útil quando o suporte de hierarquia estiver sobrecarregado. Neste caso, presume-se que a hierarquia seja superficial e que existem apenas contêineres e seus registros filhos (periódicos e artigos). Na prática, o suporte ao link simples busca dois objetivos:

1. Substituir a ligação de um registro filho por contêiner por título com linkagem utilizando ID de registro.
2. Exibir o número de registros filhos na página de registro do contêiner com um link para localizá-los.

Vale salientar que ligações simples requerem apenas um subconjunto de campos de hierarquia, onde:

1. O registro do contêiner deve ter *is_hierarchy_id*.
2. Os registros filhos devem ter *hierarchy_parent_id*.
3. Adicionalmente, os registros filhos devem ter *container_title* para apontar o contêiner a ser exibido.
4. Registros filhos também podem ter *container_reference* que é exibido após o link e pode incluir, por exemplo, emissão e número da página.

Caso os pré-requisitos acima sejam atendidos, os links simples podem ser ativados configurando `simpleContainerLinks` como *true* na seção [Hierarchy] do arquivo **HOME_VUFIND/local/vufind/config.ini**.

¹¹ Para mais informações sobre a configuração consulte a documentação oficial no página oficial do VuFind, disponível em:< https://vufind.org/wiki/indexing:hierarchies_and_collections>.

O modo de pesquisa de coleção do VuFind também oferece uma visão de alto nível de todos os registros configurados para serem tratados como coleções. Para habilitar a visualização de coleções, é necessário definir a configuração de coleções na seção [Collections] do config.ini para *true*, conforme descrito abaixo:

A visão de coleção é um híbrido entre uma visualização de registro e uma tela de resultados de pesquisa, podendo-se personalizar essa exibição usando módulos de recomendação como qualquer outra pesquisa. As opções de configuração relevantes podem ser encontradas no arquivo Collection.ini, que fica no mesmo diretório que o arquivo config.ini. Outras opções de configuração podem ser encontradas no arquivo **HierarchyDefault.ini** na mesma pasta que config.ini. A opção mais importante é *link_type* na seção [Collections], que controla quais registros são tratados como coleções, em vez de registros simples, podendo-se encaixar dentre as Três opções possíveis:

- *Top*- Apenas registros no topo de uma árvore de hierarquia são tratados como coleções. Todos os nós abaixo do topo são apresentados como uma única lista de resultados dentro da coleção.
- *All*- Qualquer registro marcado como um nodo dentro da hierarquia (ou seja, com *is_hierarchy_id* definido no índice do Solr) será tratado como uma coleção. Ao visualizar a coleção, apenas os registros imediatamente abaixo deste nodo ficarão visíveis.
- *None*- Isso desativa coleções para todos os registros usando esse driver de hierarquia. Se você tiver apenas um driver de hierarquia (a situação mais comum), será mais fácil desativar as coleções globalmente no config.ini. Essa configuração existe no caso de ser necessária em cenários mais complexos de multi-driver.

Além da visualização de coleção personalizada, o VuFind também oferece um módulo para navegar pelas listas de suas coleções. Isto pode ser encontrado em Collections/Home sob a URL base do VuFind configurada.

Para navegar por coleções, é necessário preencher o campo *hierarchy_browse* do índice no Solr. Esse campo contém um título de coleção concatenado com um ID de coleção, com um delimitador no meio. O delimitador padrão é “{{{_ID_}}}", todavia, pode-se configurar para uma string diferente na seção [Collections] do config.ini. Os exatos IDs/títulos de coleções que devem ser indexados dependem de como as coleções foram identificadas. Se o identificador de sua coleção for “Top”, então *collection_browse* deverá conter os valores de *hierarchy_top_id/hierarchy_top_title*; se o identificador da coleção for “All”,

collection_browse deverá conter os valores de *hierarchy_parent_id/hierarchy_parent_title*.

Exemplo, seja *Collection identifier*= “Top” (somente coleções de nível superior são tratadas como coleções “reais”)

```
hierarchy_top_id = 1234
hierarchy_top_title = My Collection
hierarchy_browse = My Collection{{{_ID_}}}1234
```

Collection identifier = “All” (todos os registros com filhos na hierarquia são tratados como coleções “reais”)

```
hierarchy_parent_id = 1234
hierarchy_parent_title = My Collection
hierarchy_browse = My Collection{{{_ID_}}}1234
```

Por padrão, a pesquisa de coleção usa o facetamento Solr para encontrar uma lista de coleções. No entanto, caso tenha muitas coleções, isso pode ficar lento ou ocupar muita memória. Nesse caso, pode-se alternar usando o mecanismo de navegação por título alfabético, alterando a configuração *browseType* na seção [Collections] do config.ini para “*Alphabetical*”. Ao usar essa opção, é necessário manter o banco de dados de índice atualizado, conforme descrito na Seção 4.3

4.6. Re-indexação de Índices

Conforme o VuFind se desenvolve, de tempos em tempos, ou entre versões, ou se os dados mudam com frequência, vale a pena fazer atualizações ocasionais para garantir que os índices sejam precisos. Sempre que for preciso reconstruir os índices, basta seguir estas etapas:

1. Parar o Solr:

```
$ VUFIND_HOME/solr stop
```

2. Deletar os índices no Solr:

```
$ rm -rf VUFIND_HOME/solr/biblio/index
$ rm -rf VUFIND_HOME/solr/biblio/spell*
```

OBSERVAÇÃO

Caso esteja utilizando mais de uma instância VuFind e queira deletar os índices de uma instância específica, deve-se substituir o nome “biblio” pelo nome da instância desejada.
--

3. Iniciar o solr:

<code>\$ VUFIND_HOME/solr start</code>
--

4. Importar e indexar todos os registros novamente.

Capítulo 5

Administração

Informações sobre como trabalhar com o sistema VuFind instalado, configurado e indexado, com relação às operações do dia a dia, são tratadas neste capítulo.

5.1. Operações Frequentes

- Automação- Automatizando tarefas comuns do VuFind (indexação do Solr, inicialização, etc.)
- Iniciar e parar o Solr- Como ativar e desativar o servidor de descoberta.
- Configurações PHP- Notas sobre as configurações do PHP que podem afetar o VuFind.

A seguir, será descrita cada uma das operações citadas acima.

5.1.1. Automação

Uma vez que o VuFind está executando, ainda é preciso fazer algum trabalho para mantê-lo atualizado e estável. Os detalhes exatos da automação do VuFind variam significativamente com base no seu sistema operacional. No entanto, esses são os objetivos gerais comuns:

- Atualizar os índices do VuFind com os últimos registros alterados e, ou adicionados.
- Remover registros deletados do índice de VuFind. O VuFind é empacotado com ferramentas no diretório **VUFIND_HOME/util/** que podem ajudar. Para entender como cada ferramenta trabalha, basta executar `php` juntamente com o nome da ferramenta e serão dados todos os detalhes do processo de execução.
- Certificar-se de que os índices no Solr sejam regularmente otimizados para uso mínimo de espaço e melhor desempenho. A otimização pode ser uma boa maneira de garantir que todas as alterações não confirmadas se tornem visíveis. A ferramenta “optimize.php” em **VAR_HOME/util/** é incumbida de otimizar os índices.
- *Dump*¹² do banco de dados para fins de *backup*.
- Reiniciar o Solr periodicamente. Funcionamento por períodos extremamente longos, como qualquer software complexo, pode se tornar instável; tal passo é visto mais como medida de precaução.

Para automatizar tarefas no Linux, é necessário utilizar o `crontab`. Isso permite agendar programas para serem executados em intervalos específicos, como usuário *root* ou como um usuário específico. Os detalhes do uso do `crontab` estão além do escopo deste livro, todavia existem muitos recursos úteis disponíveis em outros na web que podem ser encontrados sem maiores dificuldades.

O exemplo a seguir é baseado nas operações rotineiras especificadas acima. Inicialmente, é necessário abrir o `crontab` no terminal, logado como *root*:

```
crontab -e
```

A **flag** é específica ao sistema operacional que caso o arquivo não exista, deve-se criar tal arquivo. Uma vez que se tenha o *crontab* aberto, acrescente as rotinas a serem executadas:

¹² Realização de uma cópia do banco de dados utilizado na aplicação. Tal cópia é feita, utilizando-se ferramentas específicas do sistema de gerenciamento de banco de dados utilizado.


```
0 0 * * * usuário VUFIND_HOME/harvest/harvest_oai.php
0 1 * * * usuário VUFIND_HOME/harvest/batch-import-xsl.sh ../
local/harvest/Dir ../import/arquivo.properties
0 2 * * * usuário VUFIND_HOME/solr.sh restart
0 3 * * * usuário VUFIND_HOME/utils/optimize.php
0 4 * * * usuário mysqldump nome-do-banco > VUFIND/nome-do-
arquivo.sql -u usuário-do-banco -psenha
```

No arquivo acima, está designado que, a partir das 0 horas do próximo dia, a primeira rotina será executada (...harvest_oai.php) e as demais rotinas uma hora após a execução de cada rotina. Na parte onde é designada a importação dos registros, é necessário fornecer o diretório (“Dir”) onde será realizada a importação bem como o nome do **arquivo.properties** responsável pelas configurações e pelo apontamento para o arquivo de normalização. As rotinas a serem executadas precisam estar de posse do usuário ao qual o VuFind foi implantado. Na quarta rotina, relacionada ao backup do banco de dados, **nome-do-banco**, “usuário-do-banco” e “senha” necessitam ser os mesmos configurados no VuFind, **nome-do-arquivo.sql** à escolha do informático. Caso necessite consultar as informações sobre o banco de dados, pode-se verificar o arquivo **config.ini** em:

```
$ VUFIND_HOME/local/config/vufind/config.ini
```

5.1.2. Iniciar e Parar o Solr

Uma vez instalado o VuFind, ele só responderá às solicitações de pesquisa se o back-end do Solr estiver ativo (a menos que esteja sendo utilizado um serviço de descoberta de terceiros em vez de um índice local, caso em que se pode desconsiderar esta seção inteiramente). Esta seção explica como ativar e desativar o Solr e como automatizar o processo.

Para iniciar o Solr no Linux, basta mudar para o diretório onde se implantou o VuFind (por exemplo, VUFIND_HOME) e executar o comando abaixo:

```
$ VUFIND_HOME/solr.sh start
```

Observação

Caso o comando acima não funcione, certifique-se de que o script esteja sendo executado com o usuário designado, que as variáveis de ambientes do VuFind estejam apontadas corretamente. Por fim, que o solr.sh tenha permissões de execução:

```
$ ls -la VUFIND_HOME/solr.sh
$ source /etc/profile.d/vufind.sh
$ chmod +x VUFIND_HOME/solr.sh
```

Em seguida, necessita-se iniciar o solr.

Por questões de simplicidade, faz-se necessário configurar um serviço para iniciar ou parar o Solr, sem precisar apontar a localização correta do script no terminal. Para isso, pode-se resolver tal questão com um arquivo *systemd*. Crie um arquivo preservando a estrutura hierárquica em */etc/systemd/system/vufind.service*, e, posteriormente, adicione as seguintes linhas:

```
Description=VuFind Starter
After=network.target httpd.service

[Service]
Type=forking
ExecStart=/bin/sh -l -c 'VUFIND_HOME/solr.sh start' -x
PIDFile=VUFIND_HOME/solr/vendor/bin/solr-8080.pid
User=usuário
ExecStop=/bin/sh -l -c "VUFIND_HOME/solr.sh stop" -x
SuccessExitStatus=0
LimitNOFILE=65000
LimitNPROC=65000
[Install]
WantedBy=multi-user.target
```

Deve-se atentar à variável “User” que precisa estar preenchida com o usuário do sistema operacional no qual o VuFind tenha sido implantado.

Para habilitar o serviço, execute o comando abaixo no terminal como root:

```
systemctl enable vufind
```

Uma vez configurado com o systemd, o Solr pode ser gerenciado usando o comando `systemctl`.

```
systemctl [enable, start, stop, status] vufind
```

5.1.3. Configurações PHP

Existem várias configurações do PHP que podem afetar o VuFind de diferentes maneiras. Muitas das configurações podem ser definidas no **httpd-vufind.conf** (nem todas, embora), mas se o servidor não executar outra aplicação, ou seja, se for um servidor dedicado, tais configurações podem ser colocadas no `php.ini`. Abaixo, são citadas as principais flags que podem ser úteis ao responsável por gerenciar o VuFind:

- *memory_limit*- Esta é uma das configurações mais relevantes. Por exemplo, caso utilize suporte de hierarquia no VuFind, necessita-se aumentar o limite de memória. 512M é uma quantidade suficiente para quase tudo.
- *session.use_strict_mode*- A documentação do PHP diz: “Nota: Ativar *session.use_strict_mode* é obrigatório para a segurança geral da sessão. Todos os sites são aconselhados a habilitar tal flag.”
- *session.gc_divisor*- Em uma aplicação com alta demanda, o valor padrão de 1.000 pode fazer com que a coleta de lixo da sessão seja executada frequentemente. Aumentando-o por exemplo para 10.000 diminuirá a coleta.
- *expose_php*- O VuFind é executado em PHP, porém, por questões de segurança, não há necessidade de expor isso em todas as respostas do servidor Web. Dessa forma, é recomendável ativar tal configuração.
- *max_input_vars*- O padrão é 1.000 e, embora seja suficiente para a maioria das situações, pode ser necessário aumentar, por exemplo, para “Renew All”, para funcionar, caso os usuários tenham mais de 1.000 variáveis.
- *error_log*- Localização do log de erros do PHP. A Configuração num caminho de arquivo específico faz com que o PHP grave todos os avisos e erros no arquivo em vez do log do Apache.

5.2. Integração

Geralmente, é útil ter modelos de link que permitam que outros sistemas (como ambientes de compartilhamento de recursos, bancos de dados de terceiros etc.) se vinculem ao conteúdo dentro de uma instância do VuFind. Felizmente, as URLs do VuFind são estruturadas de uma maneira que torna isso bastante simples. Todos os links de registro devem estar no formato:

```
http: // [seu-vufind-server] / [seu-vufind-path] / Record /  
[record-id]
```

Em relação à pesquisa, todos os links devem estar no formato:

```
http: // [seu-vufind-server] / [caminho-do-seu-vufind] / Pesquisar  
/ Resultados? Type = [tipo-de-pesquisa] & lookfor = [query].
```

Valores válidos para [tipo-de-pesquisa] podem ser encontrados consultando o arquivo de configuração `searchspecs.yaml` - ou, para uma abordagem mais simples, basta realizar uma pesquisa do tipo desejado e copiar e colar o URL resultante. O arquivo `searchspecs.yaml` está localizado em:

```
$ VUFIND_HOME/local/vufind/
```

Além do conteúdo acima, pode-se facilmente realizar uma pesquisa no VuFind a partir de uma aplicação de terceiros, simplesmente, copiando o formulário de busca HTML gerado pelo VuFind. Basta seguir os seguintes passos:

1. Na página principal do VuFind, visualizar o código fonte no navegador através do serviço inspecionar.
2. Copiar o formulário que contém a caixa de pesquisa, o qual deve ser algo como:

```
<form method="get" action="/vufind/Search/Results"
name="searchForm" id="searchForm" class="search">
<label for="searchForm_lookfor"
class="offscreen">Your search terms</label>
<input id="searchForm_lookfor" type="text"
name="lookfor" size="40" value="" class="autocomplete
searcher:Solr typeSelector:searchForm_type"/>
<label for="searchForm_type" class="offscreen">Search Type</
label>
<select id="searchForm_type" name="type" data-native-
menu="false">
<option value="AllFields">All Fields</option>
<option value="Title">Title</option>
<option value="Author">Author</option>
<option value="Subject">Subject</option>
<option value="CallNumber">Call Number</option>
<option value="ISN">ISBN/ISSN</option>
<option value="tag">Tag</option>
</select>
<input type="submit" name="submit" value="Find"/>
<a href="/demo/Search/Advanced" class="small">Advanced</a>

</form>
```

1. Alterar o atributo “action” da tag <form> para incluir a URL da instância do VuFind. Por exemplo:

```
http://busca.ibict/vufind/Search/Results
```

Posteriormente, a aplicação que o código contendo o campo de pesquisa utilizado no VuFind estará apto a realizar consulta diretamente a base de dados do VuFind. Caso seja necessário forçar um determinado tipo de busca como default, pode-se adicionar elementos de entrada ocultos ao formulário no item 2.

5.3. Alterando o Número da Porta do Solr

Ao utilizar servidores onde há mais de uma aplicação executando, pode acontecer de uma porta padrão para uma dada aplicação já estar em uso por uma outra aplicação. Esta seção será dedicada a explicar como um informático pode alterar a porta utilizada pelo Solr.

Caso o Vufind utilizado seja uma versão a partir da 3.0, basta seguir os passos seguintes:

1. Parar o Solr:

```
$ VUFIND_HOME/solr.sh stop
```

2. Verificar uma porta livre para ser utilizada. No exemplo, será utilizada a porta 8081, todavia poderia ser qualquer outra porta:

```
$ fuser 8081/tcp
```

3. Editar a linha 70 no arquivo VUFIND_HOME/solr.sh:

```
$ SOLR_PORT="8081"
```

4. Editar a linha 412 no arquivo VUFIND_HOME/local/config/vufind/config.ini:

```
url= http://localhost:8081/solr
```

Iniciar o solr:

```
$ VUFIND_HOME/solr.sh start
```

Caso esteja sendo usado o VuFind 2.x ou anterior, os passos para alterar a porta são:

1. Parar o Solr:

```
$ VUFIND_HOME/solr.sh stop
```

2. Editar o arquivo VUFIND_HOME/solr/jetty/ etc/jetty.xml:

```
<Set name="port"><SystemProperty name="jetty.port"
default="8081"/></Set>
```

3. Editar o **solr.hosturl** em VUFIND_HOME/local/import/import.properties:

```
solr.hosturl = http://localhost:8081/solr/biblio/update
```

Nota

Supondo que a instância do Solr seja biblio, ou caso seja outra instância, substituir pela instância correta.

4. Editar o arquivo **VUFIND_HOME/local/vufind/config/config.ini**:

```
[Statistics]
...
solr          = http://localhost:8081/solr
...
[Index]
...
url           = http://localhost:8081/solr
```

5. Iniciar o Solr:

```
$ VUFIND_HOME/solr.sh start
```

Ao iniciar o Solr, as mudanças terão efeito.

Capítulo 6

VuFind na ENAP

O IBICT possui uma parceria de trabalho com a Escola Nacional de Administração Pública (ENAP). Entre as demandas requisitadas pela ENAP, tem-se a integração dos softwares utilizados no intuito de concentrar um único motor de busca perante todas as bases de dados mantidas pela ENAP. Diversões fatores influenciaram na escolha de uma aplicação para resolver tal demanda.

1. código aberto, ou seja, não produz custo para a entidade;
2. consolidada para resolução da demanda requisitada;
3. documentação atualizada para sanar dúvidas pertinentes que possa surgir;
4. constante evolução da aplicação.

A partir dos fatores exigidos, chegou-se ao VuFind, que consegue acoplar tudo que foi exigido do item 1 ao 4.

Dentre os softwares mantidos ou contratados pela ENAP encontra-se: DSpace, Open Journal System (OJS), Omeka, Koha, ProQuest e Ebook Central (produto provido pela ProQuest), que respondem ao protocolo OAI-PMH e serão descritos a seguir de modo conciso.

6.1. DSpace

O DSpace é o software preferido das organizações acadêmicas, sem fins lucrativos e comerciais que criam repositórios digitais abertos. De código aberto,

fácil instalação, completamente personalizável para atender às necessidades de qualquer organização.

O software tem como objetivo preservar e permitir acesso fácil e aberto a todos os tipos de conteúdo digital, incluindo texto, imagens, vídeos, pdfs etc. Além disso, conta com uma comunidade cada vez maior de desenvolvedores, seja por parte da mantenedora¹³ ou por colaboradores espalhados pelo mundo, comprometidos com a contínua expansão e melhoria do software.

Em suma, o DSpace foi criado com o propósito de proporcionar a escolha de um repositório, sem custos, que fornecesse meios para tornar as informações disponíveis de maneira aberta e de fácil gerenciamento.

6.2. OJS

O OJS é um sistema de gerenciamento e publicação de periódicos de código aberto que foi desenvolvido pelo *Public Knowledge Project*¹⁴ por meio de seus esforços para expandir e melhorar o acesso à pesquisa. Tem como objetivo tornar a publicação de acesso aberto uma opção viável para mais periódicos, no intuito de aumentar a leitura de periódicos, bem como sua contribuição para o bem público em escala global.

6.3. Omeka

Lançada em fevereiro de 2008, a Omeka¹⁵ estabeleceu-se como uma plataforma líder de publicação na Web de código aberto para coleções digitais vinculadas a museus, bibliotecas e arquivistas que desejavam publicar coleções e exposições narrativas na web com a mesma facilidade com que se poderia lançar um blog. Além disso, o Omeka trouxe consigo o conceito de que a publicação de coleções e exposições acessíveis baseadas em padrões, contendo metadados também baseados em padrões, poderia ser realizada com a criação de uma plataforma gratuita e de código aberto que oferecesse uma interface administrativa fácil de usar, contendo as principais funções de publicação de

¹³ <https://duraspace.org> site oficial da mantenedora do Dspace.

¹⁴ Site da mantenedora <https://pkp.sfu.ca/ojs/>.

¹⁵ Site oficial da mantenedora <https://omeka.org>.

coleções e arquivamento com uma arquitetura de plug-in flexível bem como uma API de tema de design avançada.

Atualmente, é utilizada por milhares de bibliotecas, museus, acadêmicos e usuários entusiastas em diversos países.

6.4. Koha

Lançado em janeiro de 2000, o Koha é um software de gestão integrada de bibliotecas que permite gerir processos administrativos da biblioteca. Tal software é acoplado com suporte a um vasto conjunto de atividades rotineiras, executadas no contexto de uma biblioteca, entre os quais se destacam:

- gestão de autoridades;
- catalogação;
- gestão de periódicos;
- circulação;
- aquisições;
- relatórios e estatísticas.

Por meio do módulo de catálogo em-linha (OPAC), é possível promover os conteúdos da biblioteca e disponibilizar uma gama de serviços aos usuários, remotamente, tais como: renovação de empréstimos, reserva de exemplares, sugestões de aquisição etc. Vale ressaltar que tal software é compatível com as principais normas internacionais da biblioteconomia, como, por exemplo: UNIMARC, MARC21, Z39.50, MARCXML, ISO2709, SRU/SRW, SIP2, RSS etc. Outro ponto positivo é que se encontra implementado em milhares de bibliotecas em todo o mundo, desde pequenas bibliotecas, a grandes bibliotecas universitárias detentoras de milhões de exemplares, apresentando robustez e completude.

6.5. ProQuest

Empresa de tecnologia provedora de conteúdo global de informação sediada em Michigan, fundada em 1938. A ProQuest fornece aplicativos e produtos

voltados a bibliotecas onde os recursos e ferramentas fornecem suporte à pesquisa, aprendizado, publicação, disseminação, aquisição, gerenciamento e descoberta de coleções de bibliotecas.

Desde a sua fundação, a empresa cresceu por meio de aquisições. Atualmente, a empresa fornece ferramentas para gerenciamento e plataformas de descoberta e citação que permitem aos usuários de biblioteca descobrir, gerenciar, usar e compartilhar pesquisas obtidas a partir de conteúdo autoritativo.

O conteúdo armazenado na sua base de dados inclui dissertações e teses, ebooks, jornais, periódicos, coleções históricas, arquivos governamentais e culturais, bem como outros bancos de dados agregados, estimado em mais de 125 bilhões de páginas digitais. O conteúdo é acessado mais comumente através de gateways de Internet de bibliotecas.

6.6. Ebook Central

Produto provido pela ProQuest que oferece uma experiência intuitiva em *e-books* para os usuários, disponibiliza o gerenciamento de descoberta, seleção, aquisição, administração e relatórios em uma única plataforma. O conteúdo mantido na base de dados conta com mais de um milhão de livros eletrônicos com conteúdo voltado a diversas áreas, inclusive, contando com obras específicas para alunos, abrangendo Educação Básica. A base de dados em constante evolução recebe aproximadamente cem mil novos títulos adicionados a cada ano.

6.7. Modelo Arquitetural

No capítulo 1, foi apresentada a arquitetura genérica do VuFind. Aqui, será apresentada uma arquitetura específica utilizada no estudo de caso em questão. Na Figura 6.1, mostram-se as bases de dados importadas dos respectivos softwares: Dspace, Ebook Central, Omeka, OJS, Koha e ProQuest.

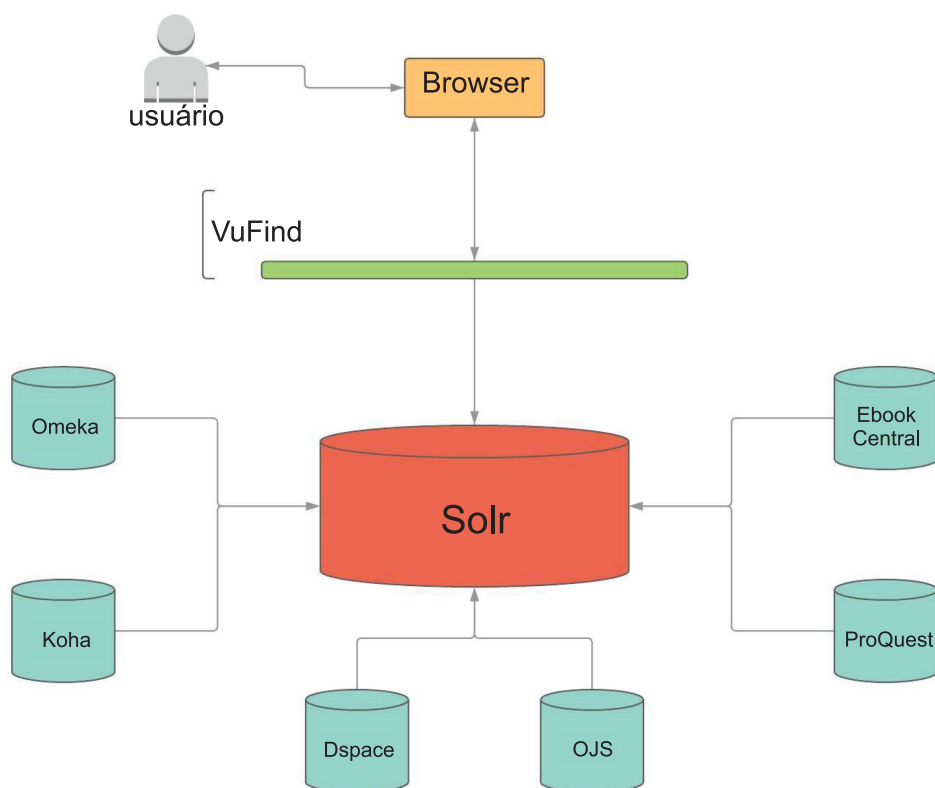


Figura 6.1 - Arquitetura VuFind na ENAP.

Fonte: Elaboração dos autores (2019).

Como forma de organização, faz-se necessário normalizar os metadados adquiridos de cada aplicação, de modo que siga um padrão para facilitar a navegação do usuário. As normalizações são aplicadas aos metadados no momento da importação dos dados e realizadas para cada software integrado sobre um conjunto fechado de metadados:

- Título;
- Autor;
- Assunto;
- Data;
- Idioma;
- Tipo de Material;
- Resumo;
- Identificador.

Tal decisão se deve ao fato que o conjunto de metadados devolvido ao coletar um registro varia de software para software.

Uma questão de suma importância é o formato dos metadados coletados. Para Dspace, OJS e Omeka, foi utilizado o formato oai_dc; por outro lado, utilizou-se o formato MARCXML para ProQuest, Ebook Central e Koha, respectivamente.

Uma vez que os softwares mantidos ou contratados na ENAP possuem uma base de dados em constante expansão, rotinas diárias de coleta e importação são adicionadas no crontab para que o VuFind proveja um conjunto de informações recuperáveis idêntico a uma consulta realizada para um dos softwares citados acima.

Para adicionar coleta e importação automática no VuFind, segue uma descrição genérica:

1. Abrir o terminal linux, em seguida logar como root e executar o seguinte comando:

```
$ crontab -e
```

2. Posteriormente, adicionar as seguintes linhas:

```
0 0 * * * usuárioap VUFIND_HOME/harvest/php harvest_oai.php
0 1 * * * usuárioap VUFIND_HOME/harvest/sh batch-import-xsl.sh
../local/harvest/BASE/ ../import/arquivo.properties
```

Observações

“0 0 * * *” e “0 1 * * *” referem-se ao horário em que serão executados os comandos de coleta e de importação no VuFind, respectivamente. Ao executar o comando “php harvest_oai.php” sem um argumento, informando em qual software deseja coletar, assume-se que todos os softwares configurados em **VUFIND_HOME/import/oai.ini** serão coletados. O argumento “usuárioap” deve ser o mesmo usuário em que o VuFind foi implantado, fazendo-se necessária a utilização do mesmo para não apresentar erros de permissões.

6.8. Integrando o Dspace ao VuFind

O repositório institucional da Enap¹⁶ é um espaço para a exposição do conteúdo intelectual produzido na ENAP, o qual inclui os programas e os materiais dos cursos, eventos, publicações, livros, trabalhos de conclusão de curso dos alunos, vídeos, relatórios etc. O conteúdo mantido é aberto ao público, ou seja, sem necessidade de qualquer cadastro, é possível fazer download, imprimir, compartilhar ou utilizar, para fins educacionais, qualquer material depositado.

Ao analisar os metadados: Título, autor e assunto; esses são armazenados conforme coletado no Dspace. Para o campo Data, optou-se por normalizar, de forma que apenas o ano seja considerado, ou seja, mês e dia são ignorados.

Exemplo: “30/12/2018” será armazenado no Vufind como “2018”.

Os **Tipos de material** (*dc.type*) existentes no Dspace são (Quadro 6.1):

Acordo Internacional	Apostila	Apresentação/Slides
Artigo	Ato Normativo	Áudio/Podcast
Autoavaliação	Avaliação	Boletim
Bibliografia	Caderno	Capítulo de Livro
Catálogo	Curso Completo	Dissertação
Editais	Ensaio	Entrevista
Estudo de Caso	Exercício/Questionário	Folders/Cartilhas
Foto/Imagem/Desenho	Infográfico/Gráfico/Tabela	Livro
Manual/Modelo	Mapa/Mapa Conceitual	Matéria Jornalística
Material Multimídia	Material Textual	Módulo Completo
Monografia/ TCC	Palestra	Periódico/Revista
Pesquisa	Plano de Ação	Programa
Projeto	Relato de Experiência	Relatório
Resenha	Tese	Texto para Discussão
Vídeo/Animação	Vídeo Aula	

Quadro 6.1 - Possíveis valores no metadado dc.type encontrados no Dspace ENAP.

Fonte: Elaboração dos autores (2019).

¹⁶ Disponível em: <<http://repositorio.enap.gov.br>>

Antes de integrar o Dspace ao VuFind, é necessário coletar os metadados provenientes do Dspace, como segue:

1. Configurar a base a ser coletada em **VUFIND_HOME/harvest/oai.ini**:

```
[DSPACE]
url = https://repositorio.enap.gov.br/oai/request
metadataPrefix = oai_dc
idSearch[] = "^oai:myuniversity.edu:/"
idReplace[] = "ds-"
idSearch[] = "\\//"
idReplace[] = "-"
injectId = "identifier"
injectDate = "datestamp"
sanitize= true
```

No campo url, tem-se a localização do Dspace da ENAP com acesso a interface oai, essa responsável por responder ao protocolo OAI_PMH.

2. Coletar os metadados via terminal no local **VUFIND_HOME/harvest/**:

```
$ php harvest_oai.php DSPACE
```

Para realizar a importação com as normalizações desejadas, é necessário implementar o arquivo **xml** que irá funcionar como um filtro, além de ligar os metadados no formato *dublin core (dc)* aos campos específicos no Solr (Quadro 6.2).

Metadados Dublin Core	Campo Solr
dc:language	language
dc:subject	topic
dc:type	format
dc:creator	author, author_sort
dc:title	title, title_short, title_full, title_sort
dc:abstract	description
dc:date	publishDate, publishDateSort
dc:identifier	url

Quadro 6.2 - Metadados dc mapeados em seus respectivos campos no Solr.
Fonte: Elaboração dos autores (2019).

Por fim, resta realizar a importação dos registros para o VuFind:

1. Navegar até o diretório de coleta:

```
$ cd VUFIND_HOME/harvest
```

1. Executar o comando de importação:

```
$ sh batch-import-xsl.sh ../local/harvest/DSPACE ../import/  
dspace.properties
```

Observações:

- “DSPACE” é o diretório onde estão armazenados os registros coletados a partir do Dspace.
- *dspace.properties* faz o apontamento para o arquivo **.xls** responsável por normalizar e ligar os metadados aos campos do Solr.
- *dspace.properties* deve ser editado em **VUFIND_HOME/import/** como segue:

```
xslt=dspace.xsl
```

Na Figura 6.2, são apresentados os metadados do item, no Dspace, na sua forma pura, ou seja, sem nenhuma normalização.

```

<repor:exportedRecords set="dspaceENAP" batchSize="1" total="3275">
  <repor:record id="oai:the.european.library.dspaceENAP:oai:repositorio.enap.gov.br:1/376" timestamp="2019-04-05">
    <repor:metadata>
      <oai_dc:dc xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai_dc/ http://www.openarchives.org/OAI/2.0/oai_dc.xsd">
        <dc:title>
          Educação a distância em organizações públicas: mesa-redonda de pesquisa-ação
        </dc:title>
        <dc:creator>Teperino, Adriana Silveira</dc:creator>
        <dc:creator>Guelfi, Antonio Carlos</dc:creator>
        <dc:contributor>Silva, Luciana Mourão Cerqueira e (Relator)</dc:contributor>
        <dc:subject>Educação a distância</dc:subject>
        <dc:subject>Capacitação Profissional</dc:subject>
        <dc:subject>Organizações públicas</dc:subject>
        <dc:subject>Educação a Distância</dc:subject>
        <dc:subject>Capacitação Profissional</dc:subject>
        <dc:subject>Setor Público</dc:subject>
      </oai_dc:dc>
    </repor:metadata>
    <dc:abstract>
      O livro materializa a quintessência dos debates travados pelo seletor grupo de representantes educadores
    </dc:abstract>
    <dc:description>Número de páginas: 200 p.</dc:description>
    <dc:description>Ensino a Distância</dc:description>
    <dc:description>ISBN 85-256-0054-7</dc:description>
    <dc:date>2006-10-12</dc:date>
    <dc:type>Livro</dc:type>
    <dc:identifier>http://repositorio.enap.gov.br/handle/1/376</dc:identifier>
    <dc:language>Idioma::Português:portuguese:pt</dc:language>
    <dc:rights>Escola Nacional de Administração Pública (ENAP)</dc:rights>
    <dc:rights>Acesso::Acesso Aberto</dc:rights>
    <dc:rights>Regime::Licença Padrão ENAP</dc:rights>
    <dc:format>application/pdf</dc:format>
    <dc:publisher>Escola Nacional de Administração Pública (ENAP)</dc:publisher>
  </repor:record>
</repor:exportedRecords>

```

Figura 6.2 - Arquivo xml coletado a partir do Dspace ENAP.

Fonte: Elaboração dos autores (2019).

Os metadados normalizados segundo a descrição da Seção 6.6, são mostrados na Figura 6.3

```
-<oai_dc:dc xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai_dc/ http://www.openarchives.org/OAI/2.0/oai_dc.xsd">
  -<dc:title>
    Educação a distância em organizações públicas: mesa-redonda de pesquisa-ação
  </dc:title>
  <dc:creator>Teperino, Adriana Silveira</dc:creator>
  <dc:creator>Guelfi, Antonio Carlos</dc:creator>
  <dc:subject>Educação a distância</dc:subject>
  <dc:subject>Capacitação Profissional</dc:subject>
  <dc:subject>Organizações públicas</dc:subject>
  <dc:subject>Educação a Distância</dc:subject>
  <dc:subject>Capacitação Profissional</dc:subject>
  <dc:subject>Setor Público</dc:subject>
  -<dc:abstract>
    O livro materializa a quintessência dos debates travados pelo seletor grupo de representantes educadores
  </dc:abstract>
  <dc:language>Português</dc:language>
  <dc:date>2006</dc:date>
  <dc:identifier>http://repositorio.enap.gov.br/handle/1/376</dc:identifier>
  <dc:type>Relatório</dc:type>
</oai_dc:dc>
```

Figura 6.3 - Metadados coletados do Dspace ENAP após a normalização.

Fonte: Elaboração dos autores (2019).

6.9. Integrando o Omeka ao VuFind

Esse é um espaço para o desenvolvimento de exposições de publicações com temas relevantes para a Administração Pública e para a sociedade. As exposições¹⁷ são fruto do trabalho de curadoria realizado pela Escola Nacional de Administração Pública como uma prática de Gestão do Conhecimento.

Em relação aos Tipos de materiais, no Omeka, em que se encontra formato *dublin core* (dc.type), os possíveis valores correspondentes ao campo são listados na Quadro 6.3. O campo data, segue o formato especificado na Seção 6.7.

Artigo	Artigo de revista	Artigo de jornal
Artigo de jornal	Apresentação	Caso de ensino
CD-ROM	Capítulo de livro	Caderno de discussão
Cartilha	Curso	Dissertação de Mestrado
Documento	Entrevista	Especialização
Evento	Folheto	Glossário
Hyperlink	Imagem	Iniciativa
Jornal	Livro	Notícia
Nota técnica	Podcast	Programa
Portfólio	Relatório	Trabalho de conclusão de curso e Tese de Doutorado

Quadro 6.3 - Possíveis valores no metadado dc.type encontrados no OmekaENAP.

Fonte: Elaboração dos autores (2019)

¹⁷ Disponível em: < <https://exposicao.enap.gov.br> >

No intuito de coletar os metadados provenientes do Omeka, é necessário configurar o arquivo **oai.ini** em **VUFIND_HOME/harvest** como segue:

```
[OMEKA]
url = https://exposicao.enap.gov.br/oai-pmh-repository/request
metadataPrefix = oai_dc
idSearch[] = "/^oai:myuniversity.edu:/"
idReplace[] = "os-"
idSearch[] = "\\/\\/\"
idReplace[] = "-\"
injectId = "identifier"
injectDate = "datestamp"
sanitize= true
```

Posteriormente, deve-se executar o comando seguinte:

```
$ php harvest_oai.php OMEKA
```

Para realizar a importação refletindo as normalizações desejadas é necessário criar e implementar o arquivo com a extensão **.xsl** em **VUFIND_HOME/import/xsl** o qual irá funcionar como um filtro e ligar os metadados no formato *dublin core* (DC) aos campos especificados na Quadro 6.2. Além disso, é necessário criar um arquivo denotado **omeka.properties**¹⁸ em **VUFIND_HOME/import**.

OBSERVAÇÃO

Pode-se copiar a partir dos exemplos presentes no diretório, por exemplo a partir do **dspace.properties**. Em seguida apontar para o arquivo com extensão **.xsl** armazenado em **VUFIND_HOME/import/xsl**, conforme o exemplo abaixo:

```
xslt=omeka.xsl
```

Após a configuração, a importação dos registros para o VuFind, é realizada como segue:

¹⁸ O nome utilizado é *omeka*, apenas por questões de padronização, todavia, o leitor pode utilizar qualquer outro de seu agrado.

1. Navegar até o diretório de coleta:

```
$ cd VUFIND_HOME/harvest
```

2. Executar o comando de importação:

```
$ sh batch-import-xsl.sh ../local/harvest/OMEKA ../import/omeka.  
properties
```

As Figuras 6.4 e 6.5 trazem o xmls com os metadados com e sem normalizações, respectivamente.

```
-<oai_dc:dc xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai_dc/ http://www.openarchives.org/OAI/2.0/oai_dc.xsd">  
  <identifier>oai:exposicao.enap.gov.br:3</identifier>  
  <timestamp>2018-11-04T17:11:33Z</timestamp>  
  <dc:title>  
    Mulheres Rurais e Autonomia: formação e articulação para efetivar políticas públicas nos Territórios da Cidadania  
  </dc:title>  
  <dc:creator>BUTTO, Andréa (Org.)</dc:creator>  
  <dc:creator>FARIA, Nalu (Org.)</dc:creator>  
  <dc:date>10/01/2014</dc:date>  
  <dc:type>livro</dc:type>  
  <dc:subject>Mulheres Rurais</dc:subject>  
  <dc:subject>Atividades Agrárias</dc:subject>  
  <dc:identifier>https://exposicao.enap.gov.br/items/show/243</dc:identifier>  
  <dc:abstract>  
    Coordenação-Geral de Mulheres Rurais, Juventude e Povos e Comunidades Tradicionais, da Secretaria Especial de Agricultura Familiar e Desenvolvimento Agrário  
  </dc:abstract>  
  <dc:language>pt_BR</dc:language>  
</oai_dc:dc>
```

Figura 6.4 - Metadados coletados no formato oai_dc a partir do Omeka ENAP.
Fonte: Elaboração dos autores (2019).

```
-<oai_dc:dc xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai_dc/ http://www.openarchives.org/OAI/2.0/oai_dc.xsd">  
  <identifier>oai:exposicao.enap.gov.br:3</identifier>  
  <timestamp>2018-11-04T17:11:33Z</timestamp>  
  <dc:title>  
    Mulheres Rurais e Autonomia: formação e articulação para efetivar políticas públicas nos Territórios da Cidadania  
  </dc:title>  
  <dc:creator>BUTTO, Andréa (Org.)</dc:creator>  
  <dc:creator>FARIA, Nalu (Org.)</dc:creator>  
  <dc:date>2014</dc:date>  
  <dc:type>livro</dc:type>  
  <dc:subject>Mulheres Rurais</dc:subject>  
  <dc:subject>Atividades Agrárias</dc:subject>  
  <dc:identifier>https://exposicao.enap.gov.br/items/show/243</dc:identifier>  
  <dc:abstract>  
    Coordenação-Geral de Mulheres Rurais, Juventude e Povos e Comunidades Tradicionais, da Secretaria Especial de Agricultura Familiar e Desenvolvimento Agrário  
  </dc:abstract>  
  <dc:language>Português</dc:language>  
</oai_dc:dc>
```

Figura 6.5 - Metadados coletados no formato oai_dc do Omeka ENAP após normalização.

Fonte: Elaboração dos autores (2019).

6.10. Integrando o OJS ao VuFind

No OJS implantado na ENAP os registros que armazenam os metadados estão no formato **oai_dc**. Com relação ao campo “tipo”, este está fixo como sendo artigo, dessa forma não foi necessário realizar mapeamento.

Coletando os registros a partir do OJS:

1. Configurado o arquivo **oai.ini** em **VUFIND_HOME/harvest**:

```
[OJS]
url = https://revista.enap.gov.br/index.php/RSP/oai
metadataPrefix = oai_dc
idSearch[] = "/^oai:myuniversity.edu:/"
idReplace[] = "ojs-"
idSearch[] = "\\/\\"
idReplace[] = "-"
injectId = "identifier"
injectDate = "datestamp"
sanitize= true
```

2. Coletar os metadados via terminal no local **VUFIND_HOME/harvest**:

```
$ php harvest_oai.php OJS
```

Após a configuração e a coleta, é necessário realizar importação dos registros para o VuFind:

3. Navegar até o diretório de coleta:

```
$ cd VUFIND_HOME/harvest
```

4. Executar o comando de importação:

```
$ sh batch-import-xsl.sh ../local/harvest/OJS ../import/ojs.
properties
```

Observações:

- “OJS” é o diretório onde foi armazenados os registros coletados a partir do OJS.

- **ojs.properties** faz o apontamento para o arquivo .xls responsável por normalizar e ligar os metadados aos campos do Solr. Para editar o arquivo, utilize um terminal para navegar até **VUFIND_HOME/import/** e configurar o arquivo **ojs.properties** como segue:

```
xslt=ojs.xsl
```

Na Figura 6.6, está apresentado um registo contendo metadados coletados na sua forma pura, sem passar por qualquer tipo de tratamento; após a normalização, os dados são armazenados conforme mostrado na Figura 6.7.

```
-<oai_dc:dc xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai_dc/ http://www.openarchives.org/OAI/2.0/oai_dc.xsd">
  -<dc:title xml:lang="pt-BR">
    Mapeamento da oferta de capacitação nas escolas de governo no Brasil: gestão da informação para fortalecimento da gestão pública
  </dc:title>
  <dc:creator>Ferrarezi, Elisabete</dc:creator>
  <dc:creator>Tomacheski, João Alberto</dc:creator>
  -<dc:abstract>
    resultados parciais do projeto "Mapeamento da Oferta de Capacitação nas Escolas de Governo".
  </dc:abstract>
  <dc:publisher xml:lang="pt-BR">Escola Nacional de Administração Pública - Enap</dc:publisher>
  <dc:date>2017-01-27</dc:date>
  <dc:type>info:eu-repo/semantics/article</dc:type>
  -<dc:identifier>
    http://revista.enap.gov.br/index.php/RSP/article/view/51
  </dc:identifier>
  -<dc:source xml:lang="pt-BR">
    Revista do Serviço Público; v. 61 n. 3 (2010); p. 287-303
  </dc:source>
  <dc:language>por</dc:language>
  -<dc:relation>
    http://revista.enap.gov.br/index.php/RSP/article/view/51/47
  </dc:relation>
</oai_dc:dc>
```

Figura 6.6 - Metadados coletados no formato oai_dc a partir do OJS ENAP.
Fonte: Elaboração dos autores (2019).

```
-<oai_dc:dc xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai_dc/ http://www.openarchives.org/OAI/2.0/oai_dc.xsd">
  -<dc:title xml:lang="pt-BR">
    Mapeamento da oferta de capacitação nas escolas de governo no Brasil: gestão da informação para fortalecimento da gestão pública
  </dc:title>
  <dc:creator>Ferrarezi, Elisabete</dc:creator>
  <dc:creator>Tomacheski, João Alberto</dc:creator>
  -<dc:abstract>
    resultados parciais do projeto "Mapeamento da Oferta de Capacitação nas Escolas de Governo".
  </dc:abstract>
  <dc:date>2017</dc:date>
  <dc:type>Artigo</dc:type>
  -<dc:identifier>
    http://revista.enap.gov.br/index.php/RSP/article/view/51
  </dc:identifier>
  <dc:language>Português</dc:language>
</oai_dc:dc>
```

Figura 6.7 - Metadados no formato oai_dc coletados do OJS ENAP após normalização.

Fonte: Elaboração dos autores (2019).

6.11. Integrando o Koha ao VuFind

Até o momento, todos os registros coletados estavam no formato **oai_dc**, todavia, para o Koha, foi escolhido o formato em **MARCXML**. Isso porque o Koha não exporta todos os registros no formato **oai_dc**. A relação dos campos mapeados nos registros coletados do Koha levam em consideração unicamente os metadados utilizados para normalização, apresentados no Quadro 6.4:

Campo em MARC21	Código em MARC21	Campo Solr
041	a	<i>language</i>
100, 110, 700	a	<i>author, author_sort</i>
245	a	<i>title, title_short, title_full, title_sort</i>
260	c	<i>publishDate, publishDateSort</i>
520	a	<i>description</i>
600	a	<i>topic</i>
942	c	<i>format</i>
999	c	<i>url</i>

Quadro 6.4 - Metadados no formato MARC21 mapeados em seus respectivos campos no Solr.

Fonte: Elaboração dos autores (2019).

O campo 942 reflete o tipo de material incluso no Koha, possuindo os seguintes atributos, conforme Quadro 6.5:

Código no Koha	Tradução
L	Livro
C	Livro Consulta
R	Livro Referência
E	Livro Especial
G	Livro Geral
VG	Vídeo
S	Periódico
T	Tese

Quadro 6.5 - Tipos de obras armazenadas no Koha.

Fonte: Elaboração dos autores (2019)

No intuito de realizar a coleta dos metadados no VuFind, deve-se configurar a origem dos registros, como a seguir:

1. Configurando-se o arquivo `oai.ini` em **VUFIND_HOME/harvest**:

```
[KOHA]
url = https://acervo.enap.gov.br/cgi-bin/koha/oai.pl
metadataPrefix = marcxml
idSearch[] = "^oai:myuniversity.edu:/"
idSearch[] = "\\//"
idReplace[] = "-"
injectId = "identifier"
injectDate = "datestamp"
sanitize= true
```

2. Coletando-se os metadados via terminal no local **VUFIND_HOME/harvest/**:

```
$ php harvest_oai.php KOHA
```

Atente-se ao parâmetro no item 1 que designa o campo “*metadataPrefix*” no mesmo. O parâmetro é preenchido com o valor **MARCXML** o que significa que os registros a serem coletados se apresentam em tal formato.

Após a configuração e a coleta, é necessário realizar importação dos registros para o VuFind:

1. Navegar até o diretório de coleta:

```
$ cd VUFIND_HOME/harvest
```

2. Executar o comando de importação:

```
$ sh batch-import-xsl.sh ../local/harvest/KOHA ../import/koha.
properties
```

Conforme comentado no decorrer deste capítulo, “KOHA” é o diretório onde foi armazenados os registros coletados a partir do Koha ENAP e “*koha.properties*” faz o apontamento para o arquivo `.xls` responsável por normalizar e

ligar os metadados aos campos do Solr. Para editar o arquivo, utilize um terminal para navegar até **VUFIND_HOME/import/** e configurar o arquivo *koha.properties* como segue:

```
xslt=koha.xsl
```

As Figura 6.8 e Figura 6.9, apresentam o conteúdo de um registro coletado no formato **MARCXML** antes e depois de passar pela normalização, respectivamente.

```
<?xml version="1.0" encoding="UTF-8" xsi:schemaLocation="http://www.loc.gov/MARC21/slim http://www.loc.gov/standards/marcxml/schema/MARC21slim.xsd">
  <record xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.loc.gov/MARC21/slim http://www.loc.gov/standards/marcxml/schema/MARC21slim.xsd">
    <controlfield tag="001">6060611212319</controlfield>
    <controlfield tag="003">OST</controlfield>
    <datafield ind1="1" ind2=" " tag="100">
      <subfield code="a">SANTOS, Reginaldo Souza</subfield>
      <subfield code="9">9479</subfield>
    </datafield>
    <datafield ind1="1" ind2="0" tag="245">
      <subfield code="a">
        Reestruturação produtiva do Estado brasileiro na perspectiva do projeto neoliberal
      </subfield>
    </datafield>
    <datafield ind1=" " ind2=" " tag="260">
      <subfield code="a">Rio de Janeiro :</subfield>
      <subfield code="b">FGV,</subfield>
      <subfield code="c">Jan. /Fev. 2004</subfield>
    </datafield>
    <datafield ind1="3" ind2=" " tag="520">
      <subfield code="a">
        Processo de reestruturação produtiva do Estado brasileiro nas décadas de 1980 e 90 tomando como referência a perspectiva do projeto neoliberal.
      </subfield>
    </datafield>
    <datafield ind1=" " ind2="4" tag="600">
      <subfield code="a">Reforma Administrativa</subfield>
      <subfield code="9">11957</subfield>
    </datafield>
    <datafield ind1=" " ind2="4" tag="650">
      <subfield code="a">Privatização</subfield>
      <subfield code="9">11967</subfield>
    </datafield>
    <datafield ind1="1" ind2=" " tag="700">
      <subfield code="a">SANTOS, Thiago Chagas Silva</subfield>
      <subfield code="9">26367</subfield>
    </datafield>
    <datafield ind1=" " ind2=" " tag="942">
      <subfield code="c">S</subfield>
    </datafield>
    <datafield ind1=" " ind2=" " tag="999">
      <subfield code="a">Convertido do Formato PHL</subfield>
      <subfield code="b">PHL2MARC21 1.1</subfield>
      <subfield code="c">17164</subfield>
    </datafield>
    <datafield ind1=" " ind2=" " tag="941">
      <subfield code="a">por</subfield>
    </datafield>
  </record>
</?xml>
```

Figura 6.8 - Metadados coletados no formato MARCXML do Koha ENAP.

Fonte: Elaboração dos autores (2019)

```

<record xsi:schemaLocation="http://www.loc.gov/MARC21/slim http://www.loc.gov/standards/marcxml/schema/MARC21slim.xsd">
  <controlfield tag="001">6060611212319</controlfield>
  <datafield ind1="1" ind2=" " tag="100">
    <subfield code="a">SANTOS, Reginaldo Souza</subfield>
    <subfield code="g">9479</subfield>
  </datafield>
  <datafield ind1="1" ind2="0" tag="245">
    <subfield code="a">
      Reestruturação produtiva do Estado brasileiro na perspectiva do projeto neoliberal
    </subfield>
  </datafield>
  <datafield ind1=" " ind2=" " tag="260">
    <subfield code="a">Rio de Janeiro </subfield>
    <subfield code="b">FGV,</subfield>
    <subfield code="c">2004</subfield>
  </datafield>
  <datafield ind1="3" ind2=" " tag="520">
    <subfield code="a">
      Processo de reestruturação produtiva do Estado brasileiro nas décadas de 1980 e 90 tomando como referência a perspectiva do projeto neoliberal.
    </subfield>
  </datafield>
  <datafield ind1=" " ind2="4" tag="600">
    <subfield code="a">Reforma Administrativa</subfield>
    <subfield code="g">11957</subfield>
  </datafield>
  <datafield ind1="1" ind2=" " tag="700">
    <subfield code="a">SANTOS, Thiago Chagas Silva</subfield>
    <subfield code="g">26367</subfield>
  </datafield>
  <datafield ind1=" " ind2=" " tag="942">
    <subfield code="c">Periódico</subfield>
  </datafield>
  >
  <datafield ind1=" " ind2=" " tag="999">
    <subfield code="a">Convertido do Formato PHL</subfield>
    <subfield code="b">PHL2MARC21 1.1</subfield>
    <subfield code="c">17164</subfield>
  </datafield>
  <datafield ind1=" " ind2=" " tag="041">
    <subfield code="a">Português</subfield>
  </datafield>
</record>

```

Figura 6.9 - Metadados coletados no formato marc xml do Koha ENAP após normalização.

Fonte: Elaboração dos autores (2019).

6.12. Integrando o Ebook Central ao VuFind

A integração da base de dados da Ebook Central ao VuFind ocorreu de uma forma diferente das demais descritas até o momento. Os registros foram fornecidos em formato **MARCXML**. Posteriormente, os registros foram importados ao Koha da ENAP. Dessa forma, os passos descritos de integração, na Seção 6.7, implicitamente, já integra a base de dados da Ebook Central.

6.13. Configurações Aplicadas ao VuFind

6.13.1. Tema Utilizado

O tema aplicado ao VuFind ENAP foi implementado, exclusivamente pelo IBICT, para atender as demandas requisitadas. Para implementação foi utilizado a mesma estrutura do VuFind original, todavia, na parte de imagens e folhas de estilos, houve mudanças significativas, conforme pode ser visto na Figura 6.10.

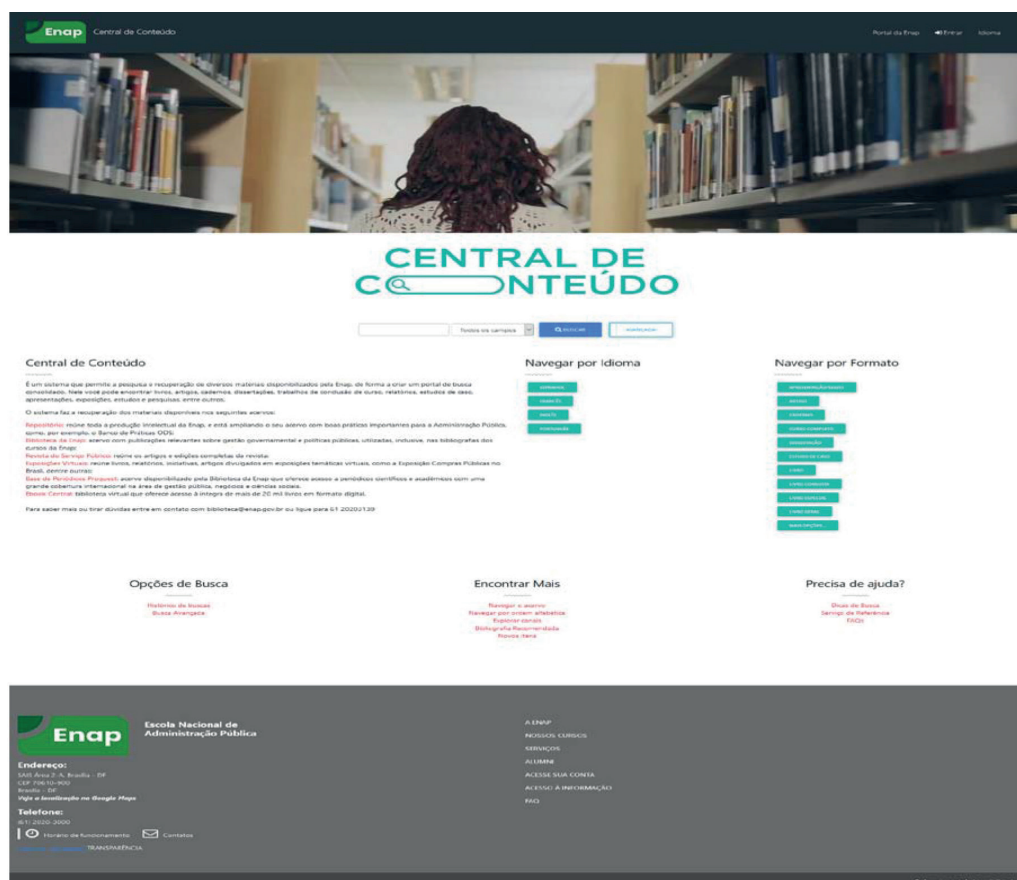


Figura 6.10 - Tema do VuFind ENAP.

Fonte: Elaboração dos autores (2019).

As técnicas de implementação utilizada, seguiu as melhores práticas designadas por *World Wide Web Consortium (W3C)*¹⁹, de forma que o

¹⁹ principal organização de padronização da *World Wide Web*, para mais detalhes consulte <http://www.w3c.br/Home/WebHome>.

conteúdo exposto se comporta conforme esperado, tanto em computador quanto smartphones, uma vez que, utilizou-se o framework *bootstrap*. Para implantar o tema, foram utilizadas as descrições fornecidas no Capítulo 2, na Seção 2.5.5.3 Tema personalizado.

A apresentação dos registros importados também foi tema discutido. Ao clicar em um item com a intenção de explorar o conteúdo, o VuFind abre uma nova página por padrão, todavia, existem outras possibilidades a serem utilizadas, como já exemplificado no Capítulo 2. O valor passado como argumento a variável “view” foi **accordion**.

The screenshot displays the VuFind interface for a book record. On the left is a sidebar with a book cover thumbnail and navigation links: 'Informação', 'Ferramentas', 'Itens Relacionados', 'Itens', and 'Descrição'. The main content area is titled 'Contabilidade pública municipal : balanços governamentais' and includes a detailed description of the book's contents, a link to 'ver descrição completa', and a table of metadata. The metadata table lists fields such as 'Autor principal', 'Formato', 'Idioma', 'Publicado em', 'Assuntos', 'Acesso em linha', and 'Tags'. The 'Formato' field has a 'Livro Geral' button next to it. The 'Assuntos' field lists several keywords related to public accounting and fiscal responsibility. The 'Acesso em linha' field provides a URL to the document's full text. The 'Tags' field is currently empty, with a prompt to add tags and a button labeled 'ADICIONAR TAG'.

Autor principal:	SILVA, Marcos Aurélio da
Formato:	Livro Geral
Idioma:	Português
Publicado em:	2007
Assuntos:	Contabilidade Pública Orçamento Público Demonstração Contábil Lei de Responsabilidade Fiscal
Acesso em linha:	https://acervo.enap.gov.br/cgi-bin/koha/opac-detail.pl?biblionumber=32284
Tags:	Sem tags, seja o primeiro a adicionar uma tag! ADICIONAR TAG

Figura 6.11 - Apresentação do conteúdo de um item.

Fonte: Elaboração dos autores (2019)

A forma de disponibilização dos itens para usuários era o fator norteador da idealização do leiaute, optou-se por um tema simples e mais comum, ou seja, com base nos outros sistemas de informação da Escola. Dessa forma, é possível atender as necessidades dos usuários e melhorar a experiência do usuário frente

a nova plataforma de recuperação da informação. Portanto, preocupou-se em manter a facilidade de aprendizado e memorização do usuário; a efetividade do sistema, isto é, o aumento de produtividade em função da curva de aprendizado; a maximização da produtividade e minimização da taxa de erros, no que tange a configuração do Portal de Conteúdo da ENAP.

6.13.2. Organização das Coleções

Uma vez que o processo de coleta e importação esteja concluído, os conjuntos de registros, importados a partir de cada aplicação, serão organizados em uma coleção no VuFind. Dessa forma, o VuFind da ENAP - Central de Conteúdo - proverá as seguintes coleções:

- Biblioteca Graciliano Ramos - *itens provenientes a partir do Koha*;
- Ebook Central - *itens provenientes a partir do Koha*;
- Exposições ENAP - *itens provenientes do Omeka*;
- Revista do Serviço Público - *itens provenientes do OJS*;
- Repositório Institucional da ENAP - *com itens provenientes do Dspace*.

Refinar a Busca	
Recursos ▲	
ENAP-Escola Nacional de Administração Pública	13,530,729
Coleção ▲	
ProQuest	13,427,791
Biblioteca Graciliano Ramos	74,901
Ebook Central	21,829
Repositório Institucional da ENAP	3,314
Revista do Serviço Público	1,950
Exposições da ENAP	944

Figura 6.12 - Coleções no VuFind ENAP.

Fonte: Elaboração dos autores (2019).

Vale ressaltar que o VuFind apresenta as coleções seguindo a quantidade de registros, onde coleções que possuem maior número de itens aparecem acima de coleções com menor número de itens (Figura 6.12).

6.13.3. Configurações de Busca

Na busca houve duas mudanças, foi retirado o **metadado Área/Cota** e incluído o **metadado Tipo**. Tal decisão se deve ao fato que, o campo Tipo possui um peso maior no filtro das buscas, o que não se apresentava ao **metadado Área/Cota**. As modificações foram realizadas conforme descrito no Capítulo 4, onde o **metadado Tipo** foi mapeado no campo Solr “**format**”.



Figura 6.13 - Alterações na busca.

Fonte: Elaboração dos autores (2019).

As customizações realizadas na página inicial de busca do VuFind (Figura 6.13 - Alterações na busca.) tem como objetivo guiar o usuário a realizar uma busca mais objetiva, permitindo mais facilidade para buscar informações, elevando a experiência do usuário. A busca avançada também possibilita recuperar informação aplicando lógica booleana e demais operadores entre os campos da busca.

Considerações finais

Atualmente, as bibliotecas gerenciam as informações presentes em seus acervos físicos e digitais. Essa situação pode dificultar encontrar a informação disponível aos seus usuários, pois precisam utilizar distintos sistemas de informação para buscar diferentes tipos de recursos informacionais, como, por exemplo: OPACs, para recursos presentes no acervo físico; repositórios institucionais, para recursos atrelados à produção acadêmico-científica e artística da instituição; bases de dados e portais de periódicos entre outros.

Nesse sentido, é necessário ações que possibilitem a integração dessas fontes, de modo que a experiência dos sujeitos se baseie em apenas uma interface de pesquisa. Nesse contexto, emergem os Sistemas de Descoberta e Entrega (SDE), que possibilitam integrar os recursos informacionais dispersos nas fontes de informação distintas, bem como reúnem os resultados das buscas sobre os recursos em uma única interface que orienta o sujeito a encontrar o que deseja.

Este livro visou apresentar as premissas teóricas, técnicas e tecnológicas que permeiam a implementação de SDE no ecossistema informacional das bibliotecas, restringindo-se ao VuFind. Assim, teve um foco maior na implementação, no que diz respeito às atividades efetuadas pela equipe de informática, com o acompanhamento técnico de profissionais de informação, embasados em teorias que garantem melhor resultados.

Com base nas pesquisas efetuadas e nos estudos aprofundados sobre o VuFind, esta obra materializa o conhecimento adquirido pela equipe de pesquisa, compartilhando as experiências e soluções encontradas no decorrer da pesquisa. Muitas informações aqui apresentadas não são mostradas em outros manuais disponíveis, visto serem resultados dos estudos efetuados durante o projeto.

Assim, torna-se uma obra única, sendo a primeira em língua vernácula, fruto da colaboração entre estudiosos da área da Ciência da Informação e Informática, que revela a complementariedade das disciplinas, revelando a fertilidade da união. Informação e informática permeiam a vida cotidiana atual, requerendo estudos que facilitem melhores formas de atuação.

Referências

BREEDING, M. **The Future of Library Resource Discovery**. Baltimore: NISO, 2015. 49 p. Disponível em: <https://groups.niso.org/apps/group_public/download.php/15077/future_library_resource_discovery.pdf>. Acesso em: 27 maio 2018.

DUBLIN CORE USAGE COMMITTEE. **DCMI Qualifiers**. jul. 2000. Disponível em: <<http://dublincore.org/documents/dcmes-qualifiers/>>. Acesso em: 08 jan. 2019.

DURASPACE. **DSpace AIP Format**. Disponível em: <<https://wiki.duraspace.org/display/DSDOC6x/DSpace+AIP+Format>>. Acesso em: 08 jan. 2019.

FOSTER, A. Determining Librarian Research Preferences: A Comparison Survey of Web-Scale Discovery Systems and Subject Databases. **The Journal of Academic Librarianship**, [s.l.], v. 44, n. 3, p.330-336, maio, 2018. Disponível em: <<http://dx.doi.org/10.1016/j.acalib.2018.04.001>>. Acesso em: 09 mar. 2019.

HOEPPNER, A. The ins and Outs of Evaluating Web-Scale Discovery Services. **Computers in Libraries**, v. 32, n. 3, p. 38–40, 2012. Disponível em: < <http://www.infotoday.com/cilmag/apr12/Hoeppner-Web-Scale-Discovery-Services.shtml>>. Acesso em: 08 jan. 2019.

MORENO, F. P.; BRASCHER, M. MARC, MARCXML E FRBR: relações encontradas na literatura. **Inf. & Soc.:Est.**, João Pessoa, v.17, n.3, p.13-25, set./dez. 2007. Disponível em: <<http://www.periodicos.ufpb.br/ojs/index.php/ies/article/view/834>>. Acesso em: 08 jan. 2019.

NATIONAL FEDERATION OF ADVANCED INFORMATION SERVICES. **NFAIS: Recommended Practices: Discovery Services**. Annapolis, 2013. 12

p. Disponível em: <https://nfais.memberclicks.net/assets/docs/BestPractices/recommended_practices_final_aug_2013.pdf>. Acesso em: 08 jan. 2019.

NATIONAL INFORMATION STANDARDS ORGANIZATION. **NISO RP-19-2014**: Open Discovery Initiative: Promoting Transparency in Discovery. Baltimore: National Information Standards Organization (NISO), 2014. Disponível em: <http://www.niso.org/apps/group_public/download.php/14820/rp-192014_ODI.pdf>. Acesso em: 08 jan. 2019.

OPEN ARCHIVES INITIATIVE PROTOCOL FOR METADATA HARVESTING. Disponível em: <<https://www.openarchives.org/OAI/openarchivesprotocol.html>>. Acesso em: 08 jan. 2019.

PAVÃO, C. M. G. **Comportamento de busca e recuperação da informação em serviços de descoberta em rede no contexto acadêmico**. 2014. 225 f. Tese (Doutorado em Comunicação e Informação)- Faculdade de Biblioteconomia e Comunicação, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2014. Disponível em: <<http://hdl.handle.net/10183/96705>>. Acesso em: 3 abr. 2019.

RIBEIRO, F. Gestão da informação / preservação da memória na era pós-custodial: um equilíbrio precário? In: 8ª MESA-REDONDA DE PRIMAVERA DO PORTO - “CONSERVAR PARA QUÊ?”, 8., 2004, Porto. **Anais...**. Porto: Faculdade de Letras da Universidade do Porto, 2005. p. 77 - 88. Disponível em: <<https://ler.letras.up.pt/uploads/ficheiros/artigo8861.PDF>>. Acesso em: 16 ago. 2019.

SHINTAKU, M. Tecnologias da Gestão da Informação. In: VECHIATO, F. et al. **Repositórios digitais: teoria e prática**. Curitiba: EDUTPR, 2017. p. 91-114. Disponível em: <<https://repositorio.ufrn.br/jspui/handle/123456789/24189>>. Acesso em: 08 jun. 2018.

SHINTAKU, M.; BRITO, R. F. de; NETO, S. C. A avaliação dos portais de revistas brasileiros implementados com o SEER/OJS por meio do levantamento da indexação pelo Latindex e SciELO. **Informação & Sociedade: Estudos**, v. 24, n. 2, 2014.

SHINTAKU, M.; VIDOTTI, S. A. B. G. Bibliotecas e repositórios no processo de publicação digital. **BIBLOS**, [S.l.], v. 30, n. 1, p. 61-80, nov. 2016. ISSN 2236-7594. Disponível em: <<https://periodicos.furg.br/biblos/article/view/5762/3993>>. Acesso em: 15 ago. 2019.

TONDING, F. J.; VANZ, S. A. de S. As bibliotecas universitárias brasileiras nas nuvens: plataformas de serviços para gerenciamento de bibliotecas. Seminário Nacional de Bibliotecas Universitárias (19.: 2016 out. 15-21: Manaus, AM). **Anais**. Manaus, AM: UFAM, 2016., 2016.

TONDING, F. J.; VANZ, S. A. de S. Plataformas de Serviços de Bibliotecas: a evolução dos sistemas para gerenciamento de bibliotecas. **Perspectivas em Ciência da Informação**, v.23, n.4, p.73-96, out./dez. 2018. Disponível em: <<http://portaldeperiodicos.eci.ufmg.br/index.php/pci/article/view/3302/2213>>. Acesso em 12 abr. 2019.

VAUGHAN, J. Web Scale Discovery Services: what and why? **Library Technology: reports**, Chicago, v. 47, n. 1, p.5-60, jan. 2011. Disponível em: <<https://journals.ala.org/index.php/ltr/article/view/4380/5065>>. Acesso em: 08 jan. 2019.

VUFIND. Disponível em: <<https://vufind.org/vufind/>>. Acesso em: 08 jan. 2019.

Sobre os Autores

Lucas Angelo da Silveira

Orcid: <http://orcid.org/0000-0002-8107-9659>

Lattes: <http://lattes.cnpq.br/9490636632029069>

Bacharel em Ciência da Computação pela Universidade Federal de Goiás (UFG), mestre em informática pelo programa de pós graduação em Informática pela Universidade de Brasília (UNB). Atualmente, doutorando em Informática pela UnB com ênfase em bioinformática de alto desempenho em estudos evolutivos entre espécies a partir dos genomas. Atua como pesquisador no Instituto Brasileiro em Ciência e Tecnologia (IBICT).

Diego José Macêdo

Lattes: <http://lattes.cnpq.br/2205539000237712>

Orcid: <http://orcid.org/0000-0002-5696-0639>

Tecnologista do Instituto Brasileiro de Informação em Ciência e Tecnologia (IBICT), bacharel em Sistemas de Informação e especialização em Engenharia de Software pela Universidade Católica de Brasília (UCB) e mestrando em Ciência da Informação pela Universidade de Brasília (UNB). Tem atuado em estudos sobre tecnologias livres para gestão da informação.

Ingrid Torres Schiessl

E-mail: ingridschiessl@gmail.com

Lattes: <http://lattes.cnpq.br/3155894540549262>

Orcid: <http://orcid.org/0000-0001-5815-2574>

Assistente de pesquisa do projeto SNJ-Ibict, apoiando a utilização do Koha na secretaria. Bibliotecária pela Universidade de Brasília (UnB) tem trabalhado na tradução do Koha para o português do Brasil e apoiado outras instituições no uso dessa ferramenta. Ajudou na implantação no Koha na SNJ e o portal de dados abertos.

Fernando Luiz Vechiato

Lattes: <http://lattes.cnpq.br/1727550604163016>

Orcid: <https://orcid.org/0000-0002-4157-740X>

Professor Adjunto do Departamento de Ciência da Informação da Universidade Federal do Rio Grande do Norte (UFRN) e Professor Colaborador do Programa de Pós-Graduação em Ciência da Informação da Universidade Estadual Paulista (UNESP - Campus de Marília). Doutor em Ciência da Informação (2013) e Mestre em Ciência da Informação (2010) pela Universidade Estadual Paulista “Júlio de Mesquita Filho” (UNESP - Campus de Marília).

Milton Shintaku

Lattes: <http://lattes.cnpq.br/K4258748Z7>

Orcid: <http://orcid.org/0000-0002-6476-4953>

Mestre e doutor em Ciência da Informação pela Universidade de Brasília (UnB), Coordenador de Articulação, Geração e Aplicação de Tecnologia (CoTec), do Instituto Brasileiro de Informação em Ciência e Tecnologia (Ibict), possuindo estudos envolvendo tecnologias para gestão da informação, comunicação científica e organizacional, além de interesses em organização e representação da informação e do conhecimento.

Nathalia Britto Pinheiro da Silva

Orcid: <https://orcid.org/0000-0002-2297-8240>

Lattes: <http://lattes.cnpq.br/3211856382571318>

Mestranda em Ciência da Informação, Linha de pesquisa: Informação e Tecnologia, do Programa de Pós-Graduação em Ciência da Informação na UNESP de Marília. Bibliotecária na Prefeitura da Estância Turística de Itú.

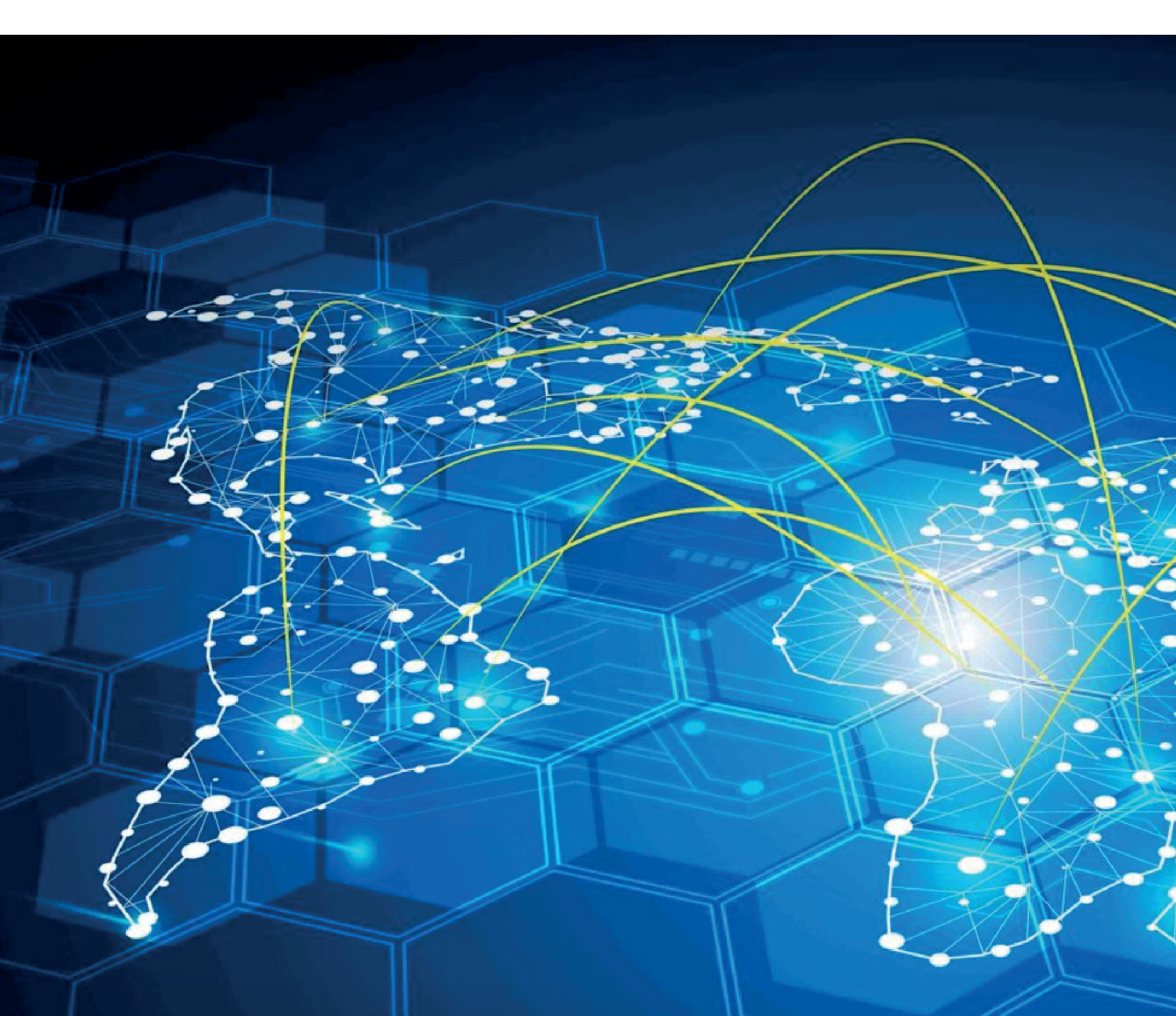
Ronnie Brito

E-mail: ronniefbrito@Ibict.br

Lattes: <http://lattes.cnpq.br/9015008667871372>

Orcid: <https://orcid.org/0000-0002-3979-603X>

Bacharel em Sistemas de Informação pela Universidade Federal de Santa Catarina, mestre e doutor em Engenharia e Gestão do Conhecimento pela Universidade Federal de Santa Catarina. Tem experiência em Mídia e Conhecimento na Educação e Acessibilidade. Atualmente é tecnologista do Instituto Brasileiro de Informação em Ciência e Tecnologia (Ibict), realizando pesquisa e desenvolvimento em sistemas para gestão da informação.



O Instituto Brasileiro de Informação em Ciência e Tecnologia (Ibict) e a Escola Nacional de Administração Pública (ENAP) trabalham juntos, na forma de pesquisas, desde 2014 com o projeto do repositório institucional. Com histórico de sucesso, essa relação tem gerado grandes resultados, em um processo colaborativo, no qual essa obra se insere. Sistemas de descoberta e entrega tem se colocado como inovação nos diversos ecossistemas de informação existentes. Assim, esse guia, fruto dessa colaboração entre ENAP e IBICT, ocupando um espaço primordial no apoio a criação da infraestrutura informacional, no que diz respeito às tecnologias apoiadas pelo IBICT e utilizadas pela por muitas instituições, incluindo a ENAP, além de promover a implementação em outras. Espera-se, assim, contribuir com as instituições na evolução e integração dos seus sistemas de informação.



Escola Nacional de
Administração Pública

MINISTÉRIO DA
ECONOMIA

MINISTÉRIO DA
CIÊNCIA, TECNOLOGIA,
INOVAÇÕES E COMUNICAÇÕES



PÁTRIA AMADA
BRASIL
GOVERNO FEDERAL

