Instituto Superiror Politécnico de Porto Amboim

MÓDULO 1
ENGENHARIA DE BIBLIOTECAS DIGITAIS

GUIA ACADÉMICO | CURSO DE ENGENHARIA INFORMÁTICA

DA ARQUITECTURA 5S À IMPLEMENTAÇÃO COM PYTHON E INTELIGÊNCIA ARTIFICIAL



Autores:

M. Sc. ALEXEI GAMBOA MOREIRA
M. Sc. GILDO DIMITRI BENTO PAULO
ENG.º HERMENEGILDO BUANDO

SUPORTE: FORMATO DIGITAL PDF

NORMA TECNOLÓGICA

RESUMO DA OBRA

Este livro-texto constitui um guia metodológico e prático para a unidade curricular de **Bibliotecas Digitais** (4.º Ano, 7.º Semestre da Licenciatura em Engenharia Informática). A obra encontra-se estruturada em quatro capítulos nucleares que cobrem o ciclo de vida completo de um sistema de informação documental:

1. **Fundamentos e Arquitectura de Bibliotecas Digitais (Capítulo 1):** Examina a evolução histórica desde as bibliotecas tradicionais até aos repositórios abertos, estabelece a arquitectura teórica formal através do *Modelo 5S* de Fox e Gonçalves, e analisa os desafios tecnológicos e sociais da preservação digital em ambientes com redes condicionadas.
2. **Gestão de Metadados e Identificadores Globais (Capítulo 2):** Detalha os princípios de interoperabilidade baseados no perfil de metadados *Dublin Core* e no protocolo de colheita *OAI-PMH*, aprofundando o uso de identificadores unívocos (ISBN, *ISSN*, DOI, *ORCID*) e a sua validação algorítmica rigorosa em código.
3. **Implementação Prática em Python para Repositórios Institucionais (Capítulo 3):** Aplica a Programação Orientada a Objectos (POO) à modelação de registos bibliográficos, desenvolve motores de busca em memória orientados a palavras-chave e implementa rotinas de serialização e persistência de dados em formatos estruturados como JSON.
4. **Inteligência Artificial, Busca Semântica e Recomendação (Capítulo 4):** Introduz a utilização de algoritmos de similaridade temática baseados no conteúdo para sugerir leituras afins, discutindo criticamente os cuidados éticos, o princípio do *Garbage In, Garbage Out* (GIGO) e a sustentabilidade de soluções tecnológicas em ambientes académicos de recursos condicionados.

Com uma abordagem pedagógica que conjuga o rigor científico, a reutilização de componentes visuais e a programação aplicada, a obra assume-se como uma ferramenta indispensável para transformar a dispersão documental num ecossistema unificado de conhecimento aberto.

Índice

O Desafio da Informação Fragmentada e o Papel Estratégico das Bibliotecas Digitais....4

Capítulo 1: Fundamentos e Arquitectura de Bibliotecas Digitais....5

1.1: Evolução Histórica e Sociedade da Informação (Da Biblioteca Física ao Repositório Digital)....5

1.2: O Modelo Teórico 5S (Streams, Structures, Spaces, Scenarios, Societies).6

1.3: Desafios Tecnológicos e Sociais da Preservação Digital em Angola....8

CAPÍTULO 2: Gestão de Metadados e Identificadores Globais....10

2.1: Estruturação de Dados e Interoperabilidade (Dublin Core e OAI-PMH)10

2.2: Identificadores Padrão (ISBN, ISSN, DOI, ORCID) e Métodos de Validação Algorítmica....12

2.2.1. Validação Algorítmica do ISBN-13 em Engenharia de Software....13

2.3: Formatos de Documentos e Acessibilidade (PDF, ePub, XML, JSON)....14

CAPÍTULO 3: Implementação Prática em Python para Repositórios Institucionais....17

3.1: Modelação Orientada a Objectos para Registos Bibliográficos....17

3.2: Motores de Pesquisa em Memória e Indexação por Palavras-Chave....19

3.2.1. Princípios da Recuperação de Informação e Indexação 19

3.2.2 Código de Implementação: Motor de Busca e Filtragem....20

3.3: Serialização e Persistência de Dados em Formato JSON/XML....22

3.3.1. O Papel da Serialização na Arquitectura de Repositórios....22

4: Inteligência Artificial, Busca Semântica e Recomendação em Bibliotecas Digitais....25

4.1: Algoritmos de Similaridade e Filtros de Recomendação por Afinidade Temática....25

Referências Bibliográficas....30

O Desafio da Informação Fragmentada e o Papel Estratégico das Bibliotecas Digitais

A sociedade contemporânea caracteriza-se por um crescimento exponencial do volume de dados gerados e disseminados através de redes informáticas. No âmbito académico e universitário, este fenómeno traduz-se numa produção incessante de monografias, teses, artigos científicos, relatórios de estágio e materiais pedagógicos. Contudo, paradoxalmente, o acesso facilitado à tecnologia tem coabitado com um problema estrutural severo: a *dispersão e fragmentação dos materiais de estudo.*

Historicamente, nas instituições de ensino superior e, em particular, em muitas realidades organizacionais de economias em desenvolvimento como a angolana, a informação académica tendeu a permanecer confinada em silos isolados. Os trabalhos de fim de curso (TCC) e as dissertações ficavam retidos em suportes físicos descentralizados ou em pastas de computadores individuais sem metadados normalizados; os suportes bibliográficos de apoio à leccionação circulavam por canais informais (como aplicações de mensagens ou pen drives); e a produção científica institucional carecia de um eixo unificador de preservação e recuperação. Esta dispersão documental gera custos cognitivos elevados para os estudantes, duplicação desnecessária de esforços de investigação e o risco iminente de perda de património intelectual.

Para responder a esta problemática, a engenharia de sistemas de informação deixa de encarar a gestão documental como uma tarefa meramente administrativa e passa a tratá-la como um *problema fundamental de arquitectura de software.* É neste cruzamento entre a Ciência da Informação e a Engenharia Informática que emergem as *Bibliotecas Digitais.*

Este manual didáctico foi concebido precisamente para dotar os futuros engenheiros informáticos das competências teóricas e práticas necessárias para conceber, desenvolver e governar repositórios institucionais resilientes. Partindo de referenciais clássicos de arquitectura (como o *Modelo 5S* (*Streams, Structures, Spaces, Scenarios, Societies*) e de padrões internacionais de interoperabilidade como o Dublin Core (norma ISO 15836), a obra transita da teoria para a prática através de implementações em *Python,* culminando na integração de algoritmos inteligentes de busca semântica e recomendação adaptados às restrições de infra-estrutura e conectividade locais.

Capítulo 1: Fundamentos e Arquitectura de Bibliotecas Digitais

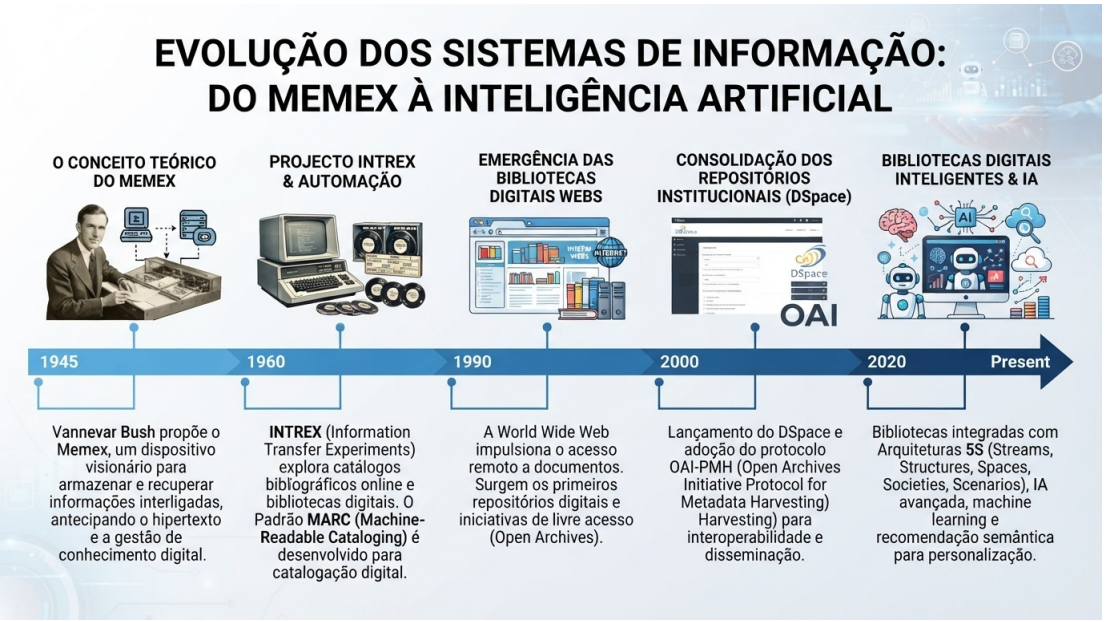
1.1: Evolução Histórica e Sociedade da Informação (Da Biblioteca Física ao Repositório Digital)

A transformação dos mecanismos de armazenamento, organização e disseminação do conhecimento científico representa uma das dinâmicas mais marcantes da História da Computação e da Ciência da Informação. Historicamente, a biblioteca física assumiu o papel de custódia central dos suportes analógicos de informação (desde os manuscritos e códices até aos livros impressos e colecções periódicas). Contudo, o crescimento exponencial da produção científica no pós-Segunda Guerra Mundial expôs os limites físicos de espaço, conservação e velocidade de acesso inerentes aos modelos tradicionais.

A génese conceptual dos sistemas modernos de recuperação de informação remonta a 1945, quando Vannevar Bush descreveu a sua visão teórica do *Memex* (Memory Extender). Bush concetualizou um dispositivo mecanizado e individual capaz de armazenar livros, registos e comunicações, permitindo a consulta rápida e a criação de redes associativas entre documentos. Esta visão pioneira antecipou a estrutura de hipertexto que viria a fundamentar a *World Wide Web* e as redes de informação digital contemporâneas.

Figura 01

Evolução Cronológica das Bibliotecas e Sistemas de Informação]



Conforme a infra-estrutura de telecomunicações e a capacidade de processamento evoluíram nas últimas décadas, o paradigma da gestão documental desdobrou-se em quatro fases e modelos coexistentes:

1. *Biblioteca Tradicional/Física***:** Centrada na gestão de acervos físicos, catalogação manual e acesso presencial local.
2. *Biblioteca Automatizada/Electrónica:* Introdução de bases de dados relacionais e catálogos de acesso público online (*OPAC - Online Public Access Catalog*), onde a busca é digital, mas a consulta do documento permanece dependente do suporte físico.
3. *Biblioteca Digital e Híbrida:* Integração de colecções nativas digitais (*born-digital*) e digitalizadas, acessíveis de forma remota através de protocolos de rede, coexistindo com acervos físicos e serviços institucionais.
4. *Repositório Institucional Aberto:* Plataforma estruturada de acesso aberto (*Open Access*) destinada a capturar, preservar e disseminar a produção intelectual e científica de uma comunidade académica e universitária.

Na transição para a Sociedade da Informação e do Conhecimento, o papel do Engenheiro Informático deixa de ser o de um mero gestor de infra-estrutura de hardware e passa a ser o de um *arquitecto de software e dados.* A concepção de uma biblioteca digital moderna exige a implementação de padrões robustos de interoperabilidade, esquemas de metadados bem definidos e motores de busca eficientes capazes de garantir a preservação a longo prazo e a descoberta célere do conhecimento.

No contexto das Instituições de Ensino Superior em Angola, a transição para bibliotecas digitais assume um carácter estratégico urgente. A desmaterialização documental não só democratiza o acesso a bibliografias científicas actualizadas para docentes e discentes, como também supera barreiras geográficas e infra-estruturais de distribuição física de livros impressos entre províncias.

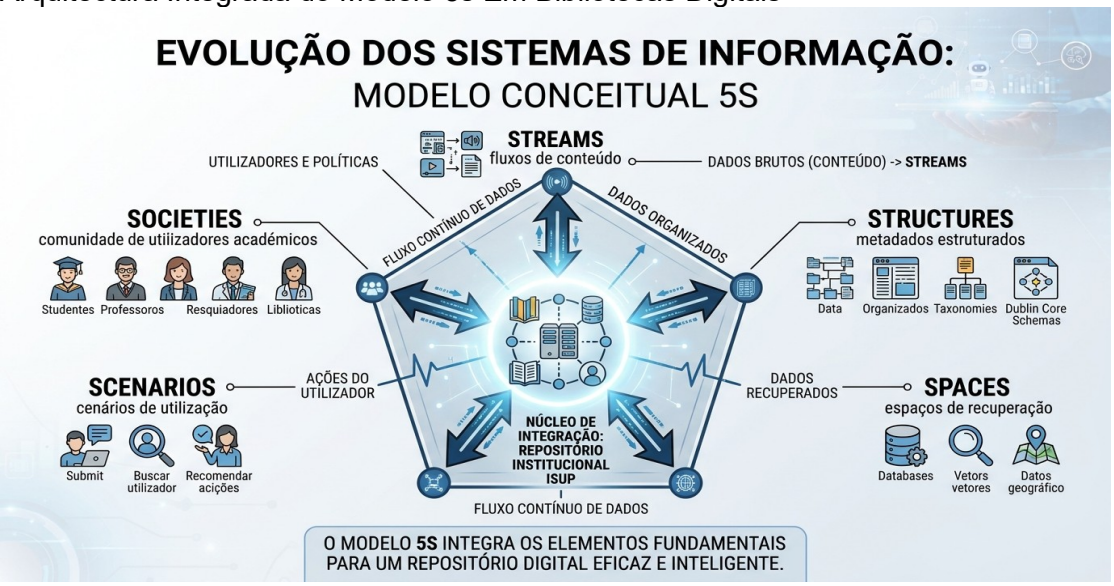
1.2: O Modelo Teórico 5S (Streams, Structures, Spaces, Scenarios, Societies)

A concepção formal e rigorosa de uma biblioteca digital exige a adopção de um modelo teórico unificador que transcenda a mera acumulação de ficheiros informáticos. Na arquitectura de sistemas de informação modernos, o referencial teórico mais consolidado e aceites internacionalmente é o *Modelo 5S*, formalizado por Edward A. Fox e Marcos André Gonçalves. Este modelo define uma biblioteca digital através de cinco construtos matemáticos e concetuais interligados: *Streams* (Fluxos), *Structures* (Estruturas), *Spaces* (Espaços), *Scenarios* (Cenários) e *Societies* (Sociedades).

Abaixo detalha-se a natureza técnica de cada um dos pilares que compõem o modelo:

1. *Streams (Fluxos):* Representam os dados brutos e o conteúdo informacional que flui na biblioteca, abrangendo desde fluxos de texto contínuo, imagens, áudio e vídeo até sequências de dados dinâmicos gerados pelos utilizadores ou por sensores IoT em ambientes académicos.
2. *Structures (Estruturas):* Constituem a organização formal da informação. Envolvem taxonomias, ontologias, hipertextos, árvores de classificação e esquemas de metadados normalizados (tais como o *Dublin Core*), permitindo estruturar as colecções para que possam ser processadas por algoritmos e motores de busca.
3. *Spaces (Espaços):* Definem os espaços multidimensionais onde os dados e os documentos são localizados e recuperados. Incluem espaços métricos, espaços vectoriais para recuperação de informação baseada em similaridade e espaços geográficos (relevantes em aplicações de Sistemas de Informação Geográfica e cartografia digital).
4. *Scenarios (Cenários):* Descrevem os serviços, as operações e os fluxos de trabalho (*workflows*) executados pelos utilizadores e pelo sistema. Um cenário especifica o comportamento da biblioteca digital perante uma acção concreta, como a submissão de uma tese de licenciatura, a validação de metadados ou a consulta de um documento por um estudante.
5. *Societies (Sociedades):* Agrupam as entidades intervenientes no ecossistema da biblioteca, englobando os utilizadores (estudantes, docentes, investigadores), os administradores de sistemas e as políticas institucionais que regem os direitos de autor, a privacidade e a governação de acesso aos acervos.

Figura 02
Arquitectura Integrada do Modelo 5s Em Bibliotecas Digitais



A aplicação do Modelo 5S ao contexto do Instituto Superior Politécnico de Porto Amboim (ISUP) garante que o desenvolvimento de plataformas como o *ISUP_Virtual* e o *SIGQ_TE* não seja conduzido de forma empírica. Ao alinhar os fluxos de documentos com estruturas normalizadas e cenários de inteligência artificial, o engenheiro informático assegura a escalabilidade, a interoperabilidade e a sustentabilidade a longo prazo do ecossistema digital institucional.

1.3: Desafios Tecnológicos e Sociais da Preservação Digital em Angola

A implementação de bibliotecas digitais em economias emergentes e, em particular, no contexto institucional de Angola, enfrenta condicionantes específicas que vão muito além da simples adopção de software de prateleira. Enquanto nas nações industrializadas o foco recai sobre a computação em nuvem em larga escala e a largura de banda ilimitada, o engenheiro informático em Angola depara-se com desafios estruturais prementes que exigem soluções de engenharia resilientes e adaptadas à realidade local.

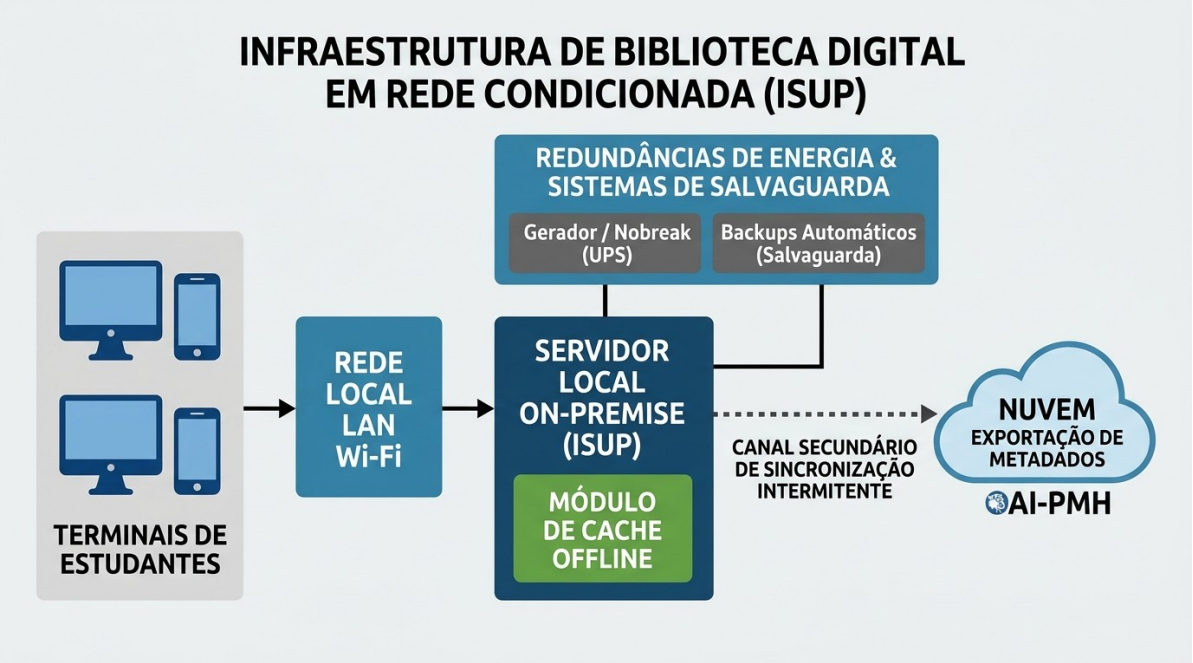
Entre os principais vectores de complexidade técnica e social destacam-se:

1. ***Conectividade de Rede Condicionada:*** A escassez ou instabilidade de ligações de alta velocidade à internet em certas regiões e instituições impõe a necessidade de arquitecturas híbridas ou locais (*on-premise*). As bibliotecas digitais devem prever mecanismos de sincronização offline, cache eficiente de metadados e optimização extrema do tamanho dos ficheiros (como documentos em PDF optimizados e formatos leves).
2. *Sustentabilidade Energética e Infra-estrutura Local***:** A dependência de redes eléctricas instáveis exige que os servidores de repositórios institucionais (como o *ISUP_Virtual*) possuam planos de contingência de energia (SOPA/UPS robustas) e políticas rigorosas de cópias de segurança (*backup*) locais e descentralizadas para evitar a perda de espólio científico.
3. *Preservação Digital a Longo Prazo:* Os formatos digitais sofrem de obsolescência tecnológica acelerada. O engenheiro deve acautelar estratégias de preservação baseadas em migração de formatos e em padrões abertos (*Open Archival Information System - OAIS*), garantindo que teses, monografias e artigos produzidos hoje permaneçam legíveis daqui a várias décadas.
4. *Capacitação e Alfabetização Digital (Societies):* A tecnologia por si só não resolve o problema do acesso. É fundamental promover a formação contínua de docentes, estudantes e investigadores para que utilizem eficazmente os repositórios,

compreendam as normas de citação, respeitem os direitos de autor e adoptem práticas de auto-regulação da aprendizagem.

FIGURA 03

Arquitectura Resiliente Para Bibliotecas Digitais em Ambientes Condicionados



A superação destes desafios constitui a verdadeira missão do engenheiro informático no ensino superior angolano: transformar barreiras infra-estruturais em oportunidades de inovação arquitectónica, garantindo que o conhecimento científico nacional seja preservado, potenciado e difundido à escala global.

CAPÍTULO 2: Gestão de Metadados e Identificadores Globais

2.1: Estruturação de Dados e Interoperabilidade (Dublin Core e OAI-PMH)

A interoperabilidade entre repositórios institucionais e bibliotecas digitais heterogéneas constitui um dos pilares fundamentais da engenharia de sistemas de informação modernos. Sem um padrão unificado de representação de metadados ("dados sobre dados"), a partilha de catálogos científicos entre universidades e bases de dados internacionais tornar-se-ia inviável devido à fragmentação sintáctica e semântica.

O referencial universalmente adoptado para a descrição normalizada de recursos digitais é o *Dublin Core Metadata Initiative (DCMI),* formalizado na norma ISO 15836. O perfil simples do *Dublin Core* assenta num conjunto reduzido e flexível de 15 elementos descritivos básicos (ver Tabela 1), concebidos para serem compreendidos tanto por seres humanos como por algoritmos de extracção automática:

Tabela 1

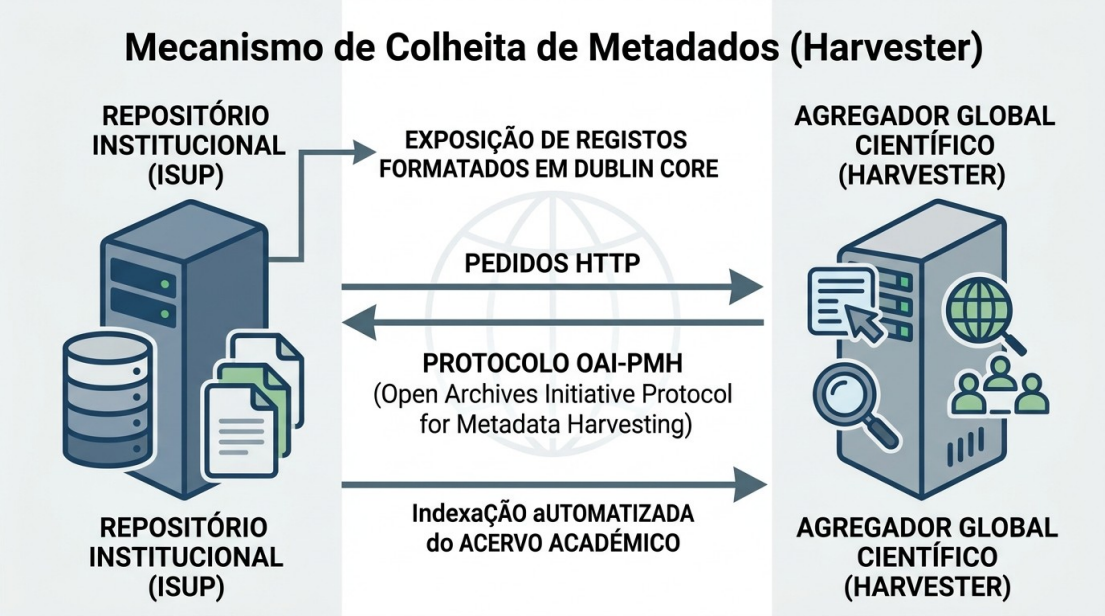
Perfil Simples do Dublin Core

Elementos	Descritivos Básicos
dc:title	O nome dado ao recurso (título da monografia, tese ou artigo).
dc:creator	A entidade principal responsável pela criação do conteúdo intelectual (autor ou investigador).
dc:subject	O tópico ou palavras-chave que descrevem o conteúdo temático da obra.
dc:description	Um resumo ou descrição textual do âmbito do documento.
dc:publisher	A entidade responsável pela disponibilização do recurso (ex.: o Instituto Superior Politécnico de Porto Amboim).
dc:contributor	Co-autores, orientadores ou entidades secundárias que colaboraram na obra.
dc:date	A data associada à criação ou disponibilização do recurso no ciclo de vida (ex.: ano de defesa do TCC).
dc:type	A natureza ou género do recurso (ex.: "Text", "Dataset", "Software").
dc:format	A manifestação física ou digital, incluindo o tipo MIME e extensão (ex.: application/pdf).

dc:identifier	Um string unívoco de identificação do recurso (ex.: ISBN, DOI ou URL permanente).
dc:source	O recurso prévio de onde o presente documento deriva, se aplicável.
dc:language	A língua em que o conteúdo intelectual está expresso (ex.: pt para Português).
dc:relation	Referência a um recurso relacionado.
dc:coverage	O âmbito espacial ou temporal da obra.
dc:rights	Informação acerca dos direitos de autor e políticas de acesso aberto (*Open Access*).

Figura 04

Esquema de Interoperabilidade e Protocolo OAI-PMH



Para além da estipulação dos campos através do *Dublin Core*, a interoperabilidade sistémica é garantida pelo protocolo *OAI-PMH (Open Archives Initiative Protocol for Metadata Harvesting).* Através de verbos estruturados em XML (como Identify, ListRecords e GetRecord), o protocolo permite que os repositórios comuniquem entre si de forma assíncrona, viabilizando a agregação da produção científica nacional em portais de âmbito regional ou internacional.

2.2: Identificadores Padrão (ISBN, ISSN, DOI, ORCID) e Métodos de Validação Algorítmica

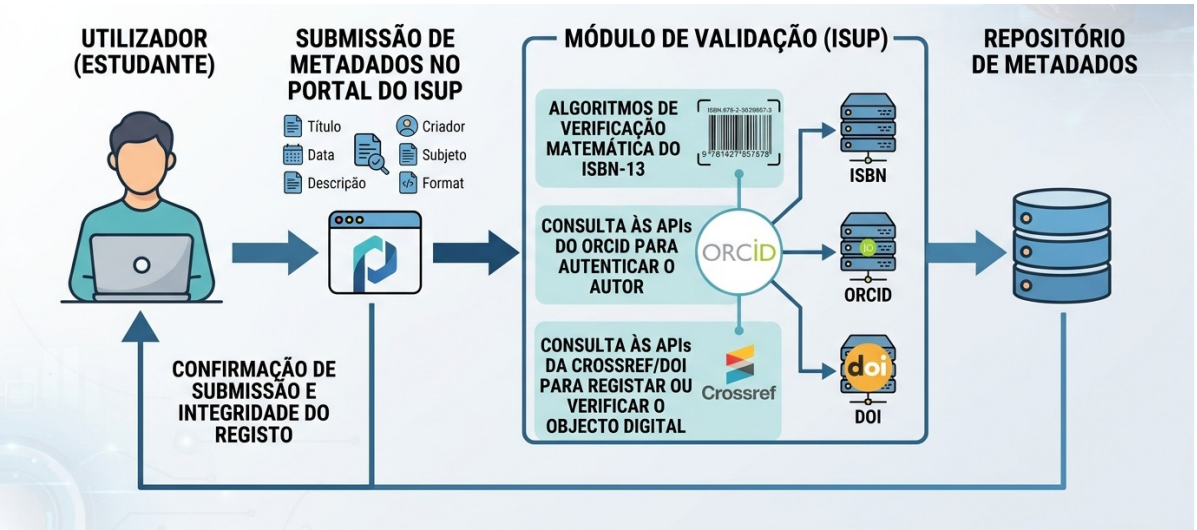
Na arquitectura de repositórios digitais e bibliotecas universitárias, a atribuição de identificadores unívocos e persistentes é indispensável para evitar duplicações, assegurar a citação científica precisa e integrar o acervo local nos ecossistemas Globais de Ciência Aberta. A utilização de strings de texto simples para representar títulos ou autores introduz ambiguidades graves decorrentes de variações ortográficas, gralhas ou homónimos.

Para resolver a ambiguidade no plano dos recursos e dos agentes, a engenharia de software recorre a quatro padrões de identificação internacionalmente consolidados:

1. *ISBN (International Standard Book Number*)**:** Identificador numérico único atribuído a livros e monografias. O formato moderno (ISBN-13), regulado pela norma ISO 2108, é composto por 13 dígitos divididos em 5 elementos: prefixo GS1 (978 ou 979), elemento de país/grupo linguístico, elemento de editora, elemento de título e dígito de controlo (*check digit*).
2. *ISSN (International Standard Serial Number):* Identificador de 8 dígitos (norma ISO 3297) concebido para a catalogação de publicações periódicas e seriadas, tais como revistas científicas, jornais e anais de conferências.
3. *DOI (Digital Object Identifier):* Identificador persistente operado pela *International DOI Foundation* e padronizado pela ISO 26324. Permite redireccionar de forma permanente o utilizador para a localização actual de um artigo ou monografia na web, através do formato URIs (https://doi.org/10.xxxx/...), eliminando o problema do *link rot* (hiperligações quebradas).
4. *ORCID (Open Researcher and Contributor ID):* Código alfanumérico não proprietário de 16 dígitos que identifica univocamente investigadores e docentes universitários, resolvendo ambiguidades relativas a nomes próprios homónimos ou mudanças na grafia de apelidos ao longo da carreira académica.

Figura 05

Arquitectura de Identificadores Persistentes e Validação de Dados



2.2.1. Validação Algorítmica do ISBN-13 em Engenharia de Software

A validação de um identificador na fase de ingestão de dados impede a contaminação do repositório por registos inválidos (Garbage In, Garbage Out). Matematicamente, a verificação da integridade de um número ISBN-13 assenta num algoritmo de ponderação alternada de pesos (1 e 3) módulo 10.

Dado um ISBN-13 expresso por x_1, x_2, ….. x_{13}, o último dígito (x_{13}) representa o dígito de controlo, o qual satisfaz a equação de congruência:

$$\left(\sum_{i=1}^{12} x_i \cdot w_i + x_{13}\right) \equiv 0 \pmod{10}$$

Onde os pesos w_i assumem o valor 1 para índices ímpares e 3 para índices pares:

$$w_i = \begin{cases} 1, & \text{se } i \text{ é ímpar} \\ 3, & \text{se } i \text{ é par} \end{cases}$$

O cálculo explícito do dígito de controlo x_{13} é obtido através da fórmula:

$$x_{13} = \left(10 - \left(\sum_{i=1}^{12} x_i \cdot w_i \pmod{10}\right)\right) \pmod{10}$$

Abaixo demonstra-se a implementação directa deste algoritmo em linguagem Python, concebida para ser integrada em módulos de validação de metadados de repositórios digitais:

```
# Módulo de Validação Algorítmica de ISBN-13
# ISUP - Engenharia de Bibliotecas Digitais
```

```
def validar_isbn13(isbn_str: str) -> bool:
    """
    Valida a integridade matemática de um código ISBN-13.
    Remove hífenes e espaços antes de aplicar o algoritmo módulo 10.
    """
    # 1. Limpeza do string de entrada
    isbn_limpo = isbn_str.replace("-", "").replace(" ", "")

    # 2. Verificação de comprimento e caracteres numéricos
    if len(isbn_limpo) != 13 or not isbn_limpo.isdigit():
        return False

    # 3. Cálculo da soma ponderada dos primeiros 12 dígitos
    soma_ponderada = 0
    for i in range(12):
        digito = int(isbn_limpo[i])
        peso = 1 if i % 2 == 0 else 3
        soma_ponderada += digito * peso

    # 4. Cálculo do dígito de controlo esperado
    digito_controlo_calculado = (10 - (soma_ponderada % 10)) % 10
    digito_controlo_fornecido = int(isbn_limpo[12])

    # 5. Comparação do dígito calculado com o dígito do ISBN
    return digito_controlo_calculado == digito_controlo_fornecido

# --- EXEMPLO DE EXECUÇÃO ---
if __name__ == "__main__":
    isbn_teste_valido = "978-989-876-543-2"
    isbn_teste_invalido = "978-989-876-543-0"

    print(f"ISBN {isbn_teste_valido} é válido? {validar_isbn13(isbn_teste_valido)}")
    print(f"ISBN {isbn_teste_invalido} é válido? {validar_isbn13(isbn_teste_invalido)}")
```

2.3: Formatos de Documentos e Acessibilidade (PDF, ePub, XML, JSON)

A escolha dos formatos de ficheiro adoptados numa biblioteca digital determina não só a qualidade da experiência de leitura do utilizador final, mas também a viabilidade da preservação digital a longo prazo e a eficiência na recuperação automatizada de dados. Um repositório institucional deve equilibrar a fidelidade visual dos documentos originais com a flexibilidade estrutural necessária para o processamento algorítmico.

No ecossistema moderno de gestão documental e engenharia de software, destacam-se quatro formatos fundamentais com papéis complementares:

1. *PDF (Portable Document Format / PDF/A):* O formato padrão para a preservação e difusão de monografias, teses e artigos em layout fixo. Para garantir a preservação a longo prazo em arquivos digitais, utiliza-se a especificação *PDF/A* (norma ISO

19005), a qual proíbe elementos dependentes de dispositivos externos (como scripts executáveis ou fontes não incorporadas), assegurando que o documento seja visualizado exactamente da mesma forma em qualquer plataforma informática, independentemente do sistema operativo.

2. *ePub (Electronic Publication):* Formato baseado em padrões abertos da W3C (*HTML5* e *CSS*) concebido especificamente para leitura adaptativa (*reflowable*). Ao contrário do PDF, o ePub redimensiona dinamicamente o texto e as imagens consoante o ecrã do dispositivo (computador, tablet ou smartphone), constituindo um elemento crítico para a acessibilidade de leitores com deficiências visuais ou condicionados por ecrãs de menor dimensão.
3. *XML (Extensible Markup Language):* Linguagem de marcação extensível utilizada para representar a estrutura lógica e semântica dos documentos e metadados. Em bibliotecas digitais, o XML permite etiquetar rigorosamente os componentes textuais (capítulos, secções, referências bibliográficas), facilitando a mineração de texto (*text mining*) e a interoperabilidade via OAI-PMH.
4. *JSON (JavaScript Object Notation):* Formato leve de intercâmbio de dados baseado em estruturas de pares chave-valor. É o padrão dominante nas APIs RESTful modernas, permitindo que os scripts em Python desenvolvidos pelos estudantes comuniquem de forma célere com o servidor e consumam metadados em tempo real nas interfaces web do repositório.

Figura 06

Ecossistema de Formatos e Acessibilidade em Repositórios Digitais]



A integração harmoniosa destes formatos assegura que o acervo académico do ISUP cumpra simultaneamente com as exigências de rigor arquivístico, acessibilidade universal para os estudantes e interoperabilidade técnica com plataformas científicas internacionais.

CAPÍTULO 3: Implementação Prática em Python para Repositórios Institucionais

3.1: Modelação Orientada a Objectos para Registos Bibliográficos

Na engenharia de software aplicada à criação de bibliotecas digitais, a transição entre o modelo conceptual de metadados (como o Dublin Core) e a sua execução num ambiente de programação exige a adopção do paradigma de Programação Orientada a Objectos (POO). Este paradigma permite encapsular os atributos descritivos de uma obra académica e os métodos operacionais de validação e manipulação dentro de abstracções reatribuíeis e reutilizáveis.

Em linguagens dinâmicas e de elevada produtividade como Python, os registos bibliográficos são modelados através de classes que traduzem os elementos essenciais do ecossistema académico do ISUP (título, autor, ano, ISBN, resumo e tipo de documento). Princípios da Modelação de Dados Bibliográficos

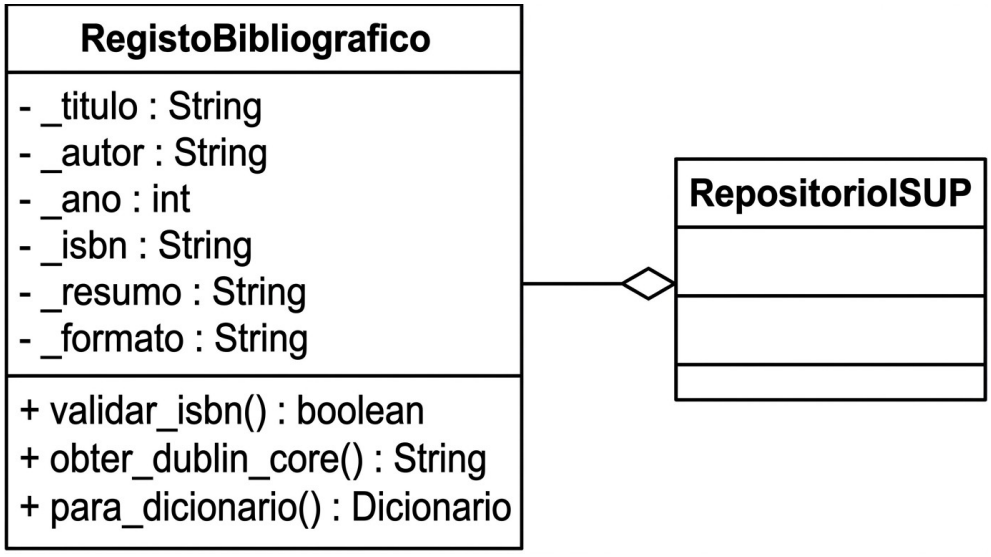
Encapsulamento e Abstracção: Os dados brutos da obra são mantidos dentro da instância da classe, expondo apenas métodos públicos para validação, formatação e conversão de dados.

Normalização dos Campos: Assegura que campos críticos (como nomes de autor ou identificadores) sejam higienizados (data cleansing) no momento da instanciação, prevenindo incoerências no banco de dados.

Mapeamento de Perfis de Metadados: Cada propriedade do objecto Python correlaciona-se directamente com uma etiqueta do padrão *Dublin Core* (dc:title, dc:creator, dc:date, dc:identifier, dc:description).

Figura 07

Diagrama de Classes Uml Para a Modelação de Obras em Python]



```
# Módulo de Modelação Orientada a Objectos - Repositório ISUP (Linguagem: Python 3)
class RegistoBibliografico:
    """
    Representa uma obra académica no repositório digital do ISUP,
    alinhada com o perfil de metadados Dublin Core.
    """
    def __init__(self, titulo: str, autor: str, ano: str, isbn: str, resumo: str, formato: str = "PDF"):
        self.titulo = titulo.strip()
        self.autor = autor.strip()
        self.ano = ano.strip()
        self.isbn = isbn.strip()
        self.resumo = resumo.strip()
        self.formato = formato.strip().upper()

    def validar_isbn(self) -> bool:
        """Valida a estrutura matemática do ISBN-13 da obra."""
        isbn_limpo = self.isbn.replace("-", "").replace(" ", "")
        if len(isbn_limpo) != 13 or not isbn_limpo.isdigit():
            return False

        soma = sum(int(digit) * (1 if idx % 2 == 0 else 3) for idx, digit in enumerate(isbn_limpo[:12]))
        digito_controlo = (10 - (soma % 10)) % 10
        return digito_controlo == int(isbn_limpo[12])

    def exportar_dublin_core(self) -> dict:
        """Retorna a representação em dicionário segundo o perfil Dublin Core."""
        return {
            "dc:title": self.titulo,
            "dc:creator": self.autor,
            "dc:date": self.ano,
            "dc:identifier": self.isbn,
            "dc:description": self.resumo,
            "dc:format": self.formato
        }

    def __repr__(self):
        return f"<Obra: '{self.titulo}' por {self.autor} ({self.ano})>"

# --- DEMONSTRAÇÃO PRÁTICA ---
if __name__ == "__main__":
    monografia = RegistoBibliografico(
        titulo="Desenvolvimento de Sistema de Rastreabilidade para Matadouros",
        autor="Eliano Gomes",
        ano="2026",
        isbn="978-989-876-543-2",
        resumo="Aplicação web para monitorização de lotes de suínos e câmaras de frio.",
        formato="PDF"
    )

    print(monografia)
    print("Metadados Dublin Core:", monografia.exportar_dublin_core())
    print("O ISBN introduzido é válido?", monografia.validar_isbn())
```

3.2: Motores de Pesquisa em Memória e Indexação por Palavras-Chave

A eficácia de uma biblioteca digital mede-se pela rapidez e precisão com que os utilizadores conseguem recuperar os documentos desejados num vasto acervo de monografias, teses e relatórios científicos. Em ambientes educacionais e de prototipagem, ou em redes locais condicionadas, a construção de um motor de busca leve em memória (*in-memory search engine*) constitui um excelente ponto de partida pedagógico e funcional para compreender os fundamentos da Recuperação de Informação (*Information Retrieval*).

3.2.1. Princípios da Recuperação de Informação e Indexação

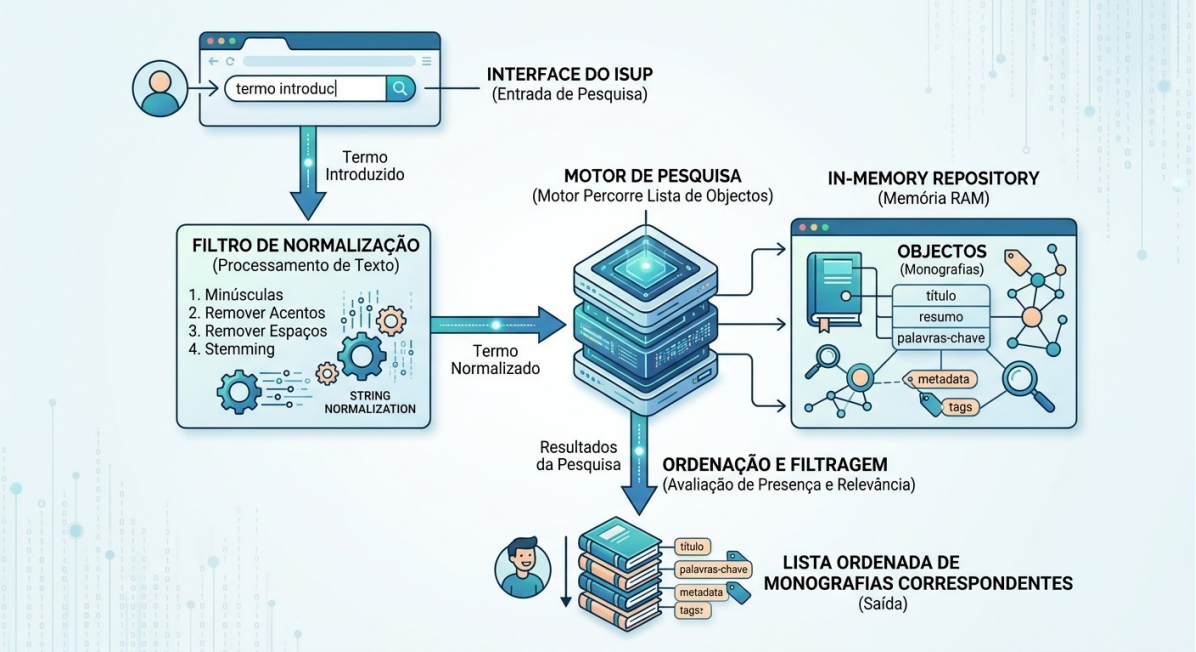
Num sistema de recuperação de informação tradicional, o motor de busca não percorre o texto integral de todos os ficheiros a cada consulta do utilizador; em vez disso, recorre a estruturas de indexação optimizadas. No contexto de objectos estruturados sob o padrão *Dublin Core*, a busca assenta tipicamente no cruzamento de termos de consulta (*query*) com os campos textuais indexados, tais como:

1. *Título (dc:title):* Onde se localizam os conceitos nucleares do trabalho académico.
2. *Resumo (dc:description):* Onde se descreve o âmbito metodológico e os resultados obtidos.
3. *Palavras-Chave (dc:subject):* O conjunto de termos normalizados que categorizam o documento.

Para implementar um motor de busca robusto em Python, o sistema deve normalizar o texto de entrada (convertendo caracteres para minúsculas e eliminando diacríticos quando necessário) para garantir que variações tipográficas não impeçam a correspondência (*string matching*).

Figura 08

Esquema de Funcionamento de um Motor de Busca em Memória



3.2.2 Código de Implementação: Motor de Busca e Filtragem

Abaixo apresenta-se o código em Python 3 que implementa a classe RepositorioDigital, capaz de armazenar instâncias de registos bibliográficos e executar buscas multi-critério por palavras-chave e autor:

```
# Módulo de Motor de Busca e Filtragem em Memória - Repositório ISUP
class RepositorioDigital:
    """
    Gere o acervo de monografias e publicações do ISUP em memória,
    permitindo operações de busca por palavra-chave e filtragem por autor.
    """
    def __init__(self):
        self.acervo = []

    def adicionar_obra(self, obra) -> None:
        """Adiciona um novo registo bibliográfico ao acervo."""
        self.acervo.append(obra)

    def pesquisar_por_palavra_chave(self, termo: str) -> list:
        """
        Procura o termo de busca nos títulos, resumos e palavras-chave
        das obras presentes no acervo. Retorna uma lista de obras correspondentes.
        """
        resultados = []
        termo_limpo = termo.strip().lower()
        if not termo_limpo:
            return resultados
        for obra in self.acervo:
            # Combinação dos campos textuais indexados
            texto_documento = f"{obra.titulo} {obra.resumo} {obra.palavra_chave}".lower()
```

```
            if termo_limpo in texto_documento:
                resultados.append(obra)
        return resultados

    def filtrar_por_autor(self, nome_autor: str) -> list:
        """Retorna todas as obras publicadas por um determinado autor ou investigador."""
        resultados = []
        autor_limpo = nome_autor.strip().lower()
        for obra in self.acervo:
            if autor_limpo in obra.autor.lower():
                resultados.append(obra)
        return resultados

# --- DEMONSTRAÇÃO PRÁTICA EM LABORATÓRIO ---
if __name__ == "__main__":
    # Inicialização do Repositório do ISUP
    repositorio = RepositorioDigital()

    # Ingestão simulada de monografias de Engenharia Informática
    from __main__ import RegistoBibliografico # Importação da classe da Epígrafe 3.1

    # (Para efeitos de demonstração autónoma, instanciamos directamente)
    class RegistoMock:
        def __init__(self, t, a, an, isbn, res, pc):
            self.titulo, self.autor, self.ano = t, a, an
            self.isbn, self.resumo, self.palavra_chave = isbn, res, pc

    repositorio.adicionar_obra(RegistoMock(
        titulo="Arquitectura de Sistemas Distribuídos em Redes Condicionadas",
        autor="Eliano Gomes",
        ano="2026",
        isbn="978-989-111-222-3",
        resumo="Aplicação web para monitorização de lotes de suínos e câmaras de frio.",
        palavra_chave="Engenharia de Software, Sistemas, Redes"
    ))

    repositorio.adicionar_obra(RegistoMock(
        titulo="Sistema Integrado de Processamento de Salários para Pessoal",
        autor="Ana Maria Caetano Kitembo",
        ano="2026",
        isbn="978-989-333-444-5",
        resumo="Plataforma para gestão de folhas de pagamento e cálculo de impostos.",
        palavra_chave="Folha de Pagamento, Algoritmos, Finanças"
    ))

    # Execução de uma pesquisa por palavra-chave
    termo = "sistemas"
    achados = repositorio.pesquisar_por_palavra_chave(termo)
    print(f"=== RESULTADOS DA BUSCA POR '{termo}' ===")
    for obra in achados:
        print(f"-> {obra.titulo} ({obra.autor})")
```

3.3: Serialização e Persistência de Dados em Formato JSON/XML

Os motores de busca e as estruturas de classes desenvolvidas em memória na epígrafe anterior são fundamentais para compreender a lógica algorítmica de uma biblioteca digital. Contudo, em sistemas reais, os objectos instanciados na memória RAM perdem-se assim que o programa termina a sua execução ou o servidor sofre um reinício. Para garantir a durabilidade das informações científicas, as monografias, teses e metadados devem ser armazenados em suportes secundários persistentes (discos rígidos ou bases de dados), um processo conhecido na engenharia de software como **serialização**.

3.3.1. O Papel da Serialização na Arquitectura de Repositórios

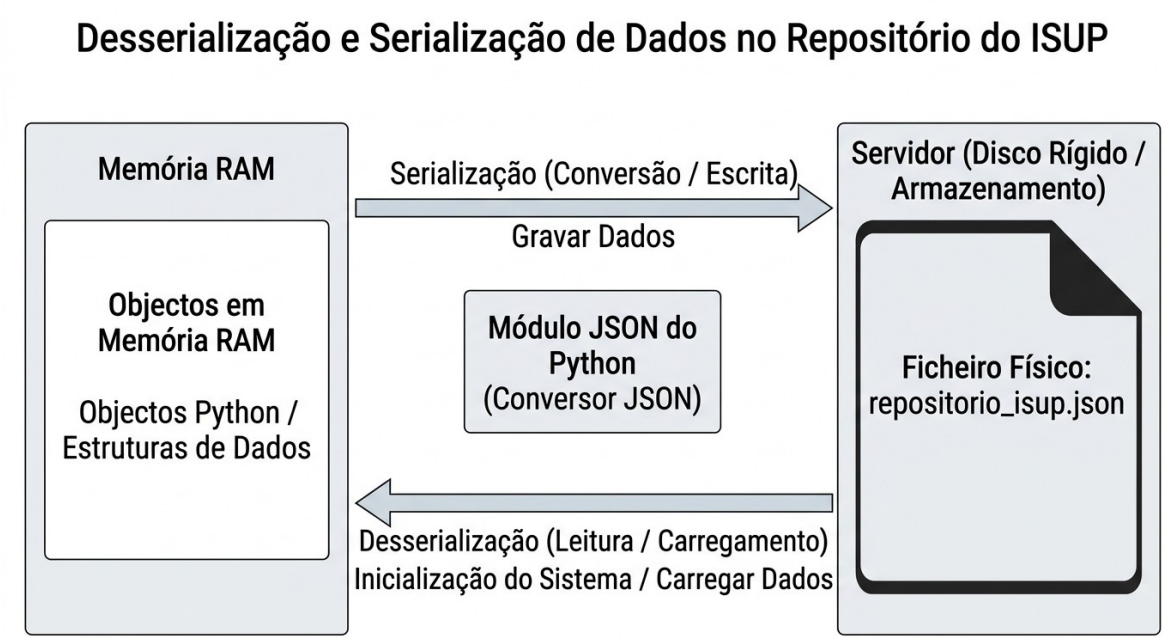
A serialização consiste no processo de traduzir a estrutura de um objecto em memória (composta por atributos e métodos) para um formato linear de fluxo de bytes ou texto que possa ser gravado num ficheiro ou transmitido através de uma rede de computadores. Posteriormente, o processo inverso (*desserialização*) reconstrói o objecto a partir do ficheiro armazenado em disco.

No ecossistema de desenvolvimento web moderno e em plataformas institucionais leves, destacam-se dois formatos principais para esta finalidade:

1. *JSON (JavaScript Object Notation):* O formato dominante para persistência local e comunicação assíncrona em APIs RESTful. A sua estrutura baseada em pares chave-valor mapeia-se de forma imediata para os dicionários e listas da linguagem Python, tornando a leitura e a escrita extremamente céleres.
2. *XML (Extensible Markup Language):* Utilizado tradicionalmente em repositórios académicos e protocolos de intercâmbio científico (como o *OAI-PMH*), o XML oferece uma validação estrutural rigorosa através de esquemas XSD (*XML Schema Definition*), garantindo a conformidade estricta com o padrão *Dublin Core*.

Figura 09.

Fluxo de Persistência e Serialização de Metadados



Código de Implementação: Persistência em JSON com Python

Abaixo apresenta-se um script em Python que estende o nosso repositório institucional, permitindo gravar todo o acervo bibliográfico num ficheiro JSON em disco e recarregá-lo automaticamente no arranque do sistema:

```
# Módulo de Serialização e Persistência - Repositório ISUP | Linguagem: Python 3

import json
from pathlib import Path

class PersistenciaRepositorio:
    """
    Gere a gravação e leitura do acervo bibliográfico do ISUP
    em formato JSON para garantir a persistência dos dados em disco.
    """
    @staticmethod
    def guardar_em_json(repositorio, caminho_ficheiro: str) -> None:
        """Serializa os registos do repositório e grava num ficheiro JSON."""
        dados_serializados = []
        for obra in repositorio.acervo:
            # Converte cada obra num dicionário compatível com Dublin Core
            registo = {
                "titulo": obra.titulo,
                "autor": obra.autor,
                "ano": obra.ano,
                "isbn": obra.isbn,
                "resumo": obra.resumo,
                "palavra_chave": getattr(obra, "palavra_chave", "")
            }
```

```
            dados_serializados.append(registo)
        # Gravação no disco com codificação UTF-8
        caminho = Path(caminho_ficheiro)
        with open(caminho, "w", encoding="utf-8") as f:
            json.dump(dados_serializados, f, ensure_ascii=False, indent=4)
                print(f"[SUCESSO] Acervo gravado com êxito em: {caminho.resolve()}")

    @staticmethod
    def carregar_de_json(repositorio, caminho_ficheiro: str) -> None:
        """Lê o ficheiro JSON em disco e reconstrói os objectos no repositório."""
        caminho = Path(caminho_ficheiro)
        if not caminho.exists():
            print(f"[AVISO] O ficheiro {caminho_ficheiro} não foi encontrado.")
            return

        with open(caminho, "r", encoding="utf-8") as f:
            dados = json.load(f)
        for item in dados:
            # Reinsere os dados no repositório institucional
            from __main__ import RegistoMock # Ou a classe oficial RegistoBibliografico
            obra_reconstruida = RegistoMock(
                t=item["titulo"],
                a=item["autor"],
                an=item["ano"],
                isbn=item["isbn"],
                resumo=item["resumo"],
                pc=item["palavra_chave"]
            )
            repositorio.adicionar_obra(obra_reconstruida)
                print(f"[SUCESSO] {len(dados)} registos carregados a partir de {caminho_ficheiro}.")


# --- DEMONSTRAÇÃO PRÁTICA EM LABORATÓRIO ---
if __name__ == "__main__":
    from __main__ import RepositorioDigital, RegistoMock

    # Inicialização
    repo = RepositorioDigital()
    repo.adicionar_obra(RegistoMock(
        titulo="Segurança em Redes de Computadores Académicos",
        autor="Daniel Vikas",
        ano="2026",
        isbn="978-989-555-666-7",
        resumo="Estudo sobre firewalls e encriptação de dados em ambientes universitários.",
        pc="Segurança, Redes, Criptografia"
    ))

    # Teste de Serialização (Gravar em disco)
    ficheiro_dados = "acervo_isup.json"
    PersistenciaRepositorio.guardar_em_json(repo, ficheiro_dados)
```

```
# Teste de Desserialização (Ler do disco para novo repositório)
repo_novo = RepositorioDigital()
PersistenciaRepositorio.carregar_de_json(repo_novo, ficheiro_dados)

print(f"O novo repositório contém {len(repo_novo.acervo)} obra(s) carregada(s) do disco.")
```

4: Inteligência Artificial, Busca Semântica e Recomendação em Bibliotecas Digitais

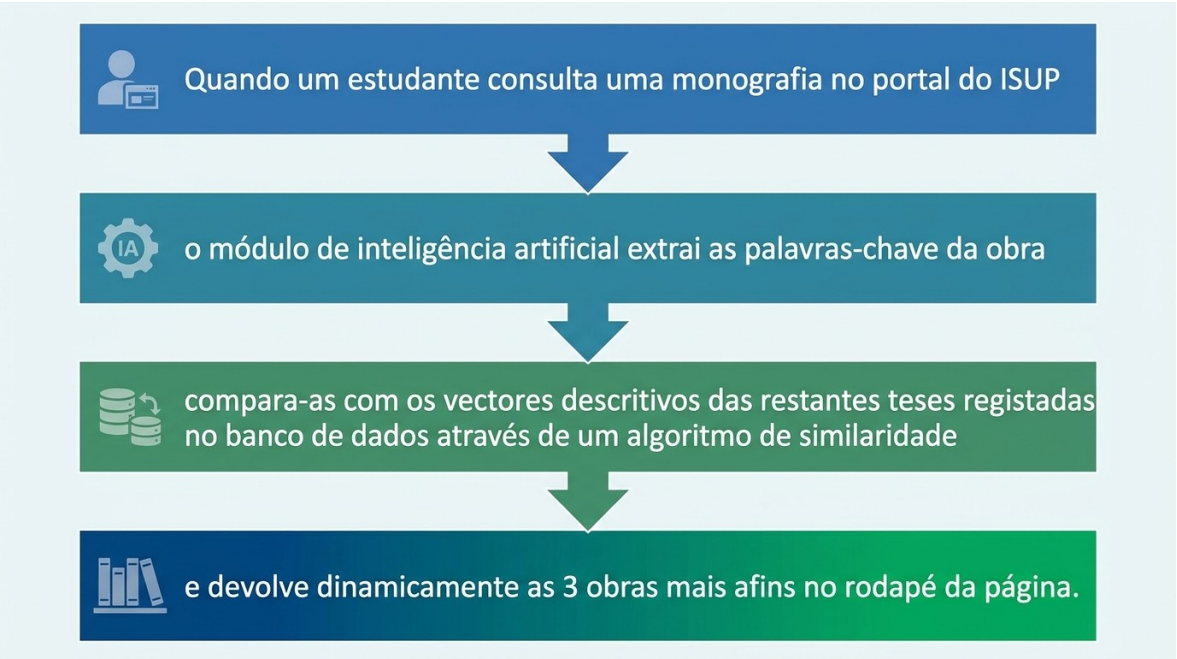
4.1: Algoritmos de Similaridade e Filtros de Recomendação por Afinidade Temática

A evolução tecnológica das bibliotecas digitais modernas exige a transição de um modelo puramente reactivo (onde o utilizador pesquisa por palavras-chave exactas e obtém resultados estáticos) para um modelo proactivo e inteligente, capaz de sugerir leituras relevantes com base nos interesses académicos e no histórico de navegação. No contexto dos repositórios institucionais, como o *ISUP_Virtual*, a integração de Sistemas de Recomendação resolve o problema da sobrecarga de informação (*information overload*), guiando estudantes e investigadores até monografias e artigos afins que talvez não descobrissem por via de uma busca convencional.

Filtragem Baseada em Conteúdo em Ambientes Académicos. Entre as diferentes abordagens de recomendação (como a filtragem colaborativa), a Filtragem Baseada em Conteúdo (Content-Based Filtering) é a mais adequada para repositórios científicos universitários. O princípio fundamental consiste em analisar os metadados descritivos das obras (títulos, resumos e palavras-chave normalizadas pelo padrão Dublin Core) e calcular a proximidade ou afinidade temática entre um documento de referência e o restante acervo. Matematicamente, a afinidade entre duas obras pode ser avaliada através da intersecção dos seus conjuntos de termos temáticos ou mediante o cálculo da Similaridade do Cosseno entre vectores de frequências de palavras. Para fins didácticos e de implementação leve em Python, podemos modelar a similaridade temática através da contagem de palavras-chave comuns nos campos de assunto (dc:subject).

Figura 10.

Arquitectura de um Sistema de Recomendação Baseado em Conteúdo



Código de Implementação: Recomendador Baseado em Afinidade Temática

Abaixo apresenta-se um script em Python que implementa um algoritmo simples e robusto de recomendação de monografias com base na sobreposição de palavras-chave temáticas:

```
# Módulo de Recomendação Baseada em Conteúdo - Repositório ISUP | Linguagem: Python 3

class MotorRecomendacao:
    """
    Algoritmo de recomendação baseado em afinidade de palavras-chave
    para monografias e artigos no repositório digital do ISUP.
    """
    def __init__(self, repositorio):
        self.repositorio = repositorio

    def recomendar_obras_similares(self, obra_referencia, limite: int = 3) -> list:
        """
        Calcula a pontuação de afinidade entre uma obra de referência
        e todas as outras obras do acervo com base nas palavras-chave.
        """
        pontuacoes = []

        # Converte as palavras-chave da obra de referência num conjunto de termos limpos
        termos_ref = set(t.strip().lower() for t in obra_referencia.palavra_chave.split(","))

        if not termos_ref:
            return []

        for obra in self.repositorio.acervo:
```

```
            # Ignora a própria obra de referência
            if obra.isbn == obra_referencia.isbn:
                continue

            termos_obra = set(t.strip().lower() for t in obra.palavra_chave.split(","))

            # Cálculo do Índice de Jaccard (intersecção sobre união de palavras-chave)
            interseccao = termos_ref.intersection(termos_obra)
            uniao = termos_ref.union(termos_obra)

            if uniao:
                coeficiente_similaridade = len(interseccao) / len(uniao)
            else:
                coeficiente_similaridade = 0.0

            if coeficiente_similaridade > 0.0:
                pontuacoes.append((obra, coeficiente_similaridade))

        # Ordena as obras por ordem decrescente de similaridade
        pontuacoes.sort(key=lambda x: x[1], reverse=True)

        # Retorna apenas os objectos das obras recomendadas até ao limite especificado
        return [item[0] for item in pontuacoes[:limite]]


# --- DEMONSTRAÇÃO PRÁTICA EM LABORATÓRIO ---
if __name__ == "__main__":
    from __main__ import RepositorioDigital, RegistoMock

    repo = RepositorioDigital()

    # Ingestão de teste de monografias do ISUP
    repo.adicionar_obra(RegistoMock(
        titulo="Desenvolvimento de Software para Matadouros",
        autor="Eliano Gomes",
        ano="2026",
        isbn="978-989-111-222-3",
        resumo="Gestão de lotes de suínos.",
        pc="Engenharia de Software, Base de Dados, Automação"
    ))

    repo.adicionar_obra(RegistoMock(
        titulo="Sistema de Folha de Pagamento Salarial",
        autor="Ana Maria Kitembo",
        ano="2026",
        isbn="978-989-333-444-5",
        resumo="Cálculo de impostos e salários.",
        pc="Algoritmos, Finanças, Base de Dados"
    ))

    repo.adicionar_obra(RegistoMock(
```

```
    titulo="Optimização de Processos em Engenharia de Software",
    autor="Estudante Investigador",
    ano="2026",
    isbn="978-989-777-888-9",
    resumo="Estudo de padrões de desenho de software.",
    pc="Engenharia de Software, Automação, Algoritmos"
))

# Teste de recomendação para a primeira obra
recomendador = MotorRecomendacao(repo)
obra_alvo = repo.acervo[0]

recomendadas = recomendador.recomendar_obras_similares(obra_alvo)
print(f"=== OBRAS RECOMENDADAS PARA '{obra_alvo.titulo}' ===")
for r in recomendadas:
    print(f"-> {r.titulo} (Autor: {r.autor})")
```

4.2: Desafios Práticos de IA em Redes Condicionadas e Cuidados Éticos (Garbage In, Garbage Out)

A integração de algoritmos de Inteligência Artificial e de motores de recomendação semântica em bibliotecas digitais abre horizontes entusiasmantes para a recuperação de conhecimento científico. Contudo, o entusiasmo tecnológico não deve encobrir os desafios operacionais, as limitações infra-estruturais e os dilemas éticos associados ao seu desenvolvimento e implementação prática, sobretudo em contextos académicos de economias emergentes como o de Angola.

O sucesso de qualquer solução baseada em IA num repositório institucional (como o *ISUP_Virtual*) está condicionado a três pilares críticos de governação de dados e engenharia:

- *O Princípio do Garbage In, Garbage Out (GIGO):* Nenhum modelo de IA, por mais sofisticado que seja, consegue extrair valor ou gerar recomendações precisas a partir de metadados corrompidos, incompletos ou mal catalogados. Se os campos descritivos (dc:title, dc:subject) submetidos pelos estudantes e investigadores contiverem erros crassos ou ausência de taxonomia normalizada, os algoritmos de similaridade falharão, gerando sugestões irrelevantes. A qualidade do dado antecede sempre a inteligência do algoritmo.
- *Restrições de Conectividade e Custo Computacional:* Modelos avançados de Processamento de Linguagem Natural (PLN) baseados em grandes redes neuronais exigem uma capacidade de processamento em nuvem e uma largura de banda de rede que muitas vezes não estão disponíveis ou implicam custos proibitivos. O engenheiro informático deve privilegiar algoritmos eficientes em memória local (como os

baseados em índices invertidos ou similaridade de conjuntos) que possam ser executados em servidores *on-premise* locais com baixo consumo energético.

- *Ética, Privacidade e Direitos de Autor:* Os sistemas inteligentes recolhem e analisam padrões de comportamento de leitura e histórico de consultas dos utilizadores. É imperativo garantir que os dados pessoais dos estudantes e docentes sejam protegidos em estrita conformidade com as leis de privacidade. Além disso, os modelos de IA não podem contornar as políticas de acesso aberto nem violar os direitos de autor das monografias e teses depositadas.

Figura 11

Governação de Dados, Ética e Resiliência na Académica



Conclusão do Manual Didáctico

A construção de bibliotecas digitais eficientes transcende a mera instalação de software comercial; exige do engenheiro informático uma visão sistémica fundamentada no Modelo 5S, normalizada pelos padrões internacionais *Dublin Core* e executada através de código limpo e resiliente em Python. Ao dominar a modelação de objectos, a persistência de dados, a busca estruturada e os princípios de inteligência artificial aplicados, os estudantes do 4.º Ano de Engenharia Informática do ISUP ficam plenamente habilitados a liderar a transformação digital e a preservação do conhecimento científico em Angola.

Referências Bibliográficas

Aggarwal, C. C. (2016). *Recommender systems: The textbook*. Springer. https://doi.org/10.1007/978-3-319-29659-5

Arms, W. Y. (2000). *Digital libraries*. MIT Press.

Baeza-Yates, R., & Ribeiro-Neto, B. (2011). *Modern information retrieval: The concepts and technology behind search* (2.ª ed.). Pearson / Addison-Wesley.

Bostrom, N., & Yudkowsky, E. (2014). The ethics of artificial intelligence. Em K. Frankish & W. M. Ramsey (Eds.), *The Cambridge handbook of artificial intelligence* (pp. 316–334). Cambridge University Press. https://doi.org/10.1017/CBO9781139046856.020

Bray, T. (2017). *The JavaScript Object Notation (JSON) data interchange format* (IETF RFC 8259). Internet Engineering Task Force. https://doi.org/10.17487/RFC8259

Bush, V. (1945). As we may think. *The Atlantic Monthly*, *176*(1), 101–108. https://www.theatlantic.com/magazine/archive/1945/07/as-we-may-think/303881/

Consultative Committee for Space Data Systems. (2012). *Reference model for an Open Archival Information System (OAIS)* (ISO Standard No. 14721:2012). International Organization for Standardization. https://www.iso.org/standard/57284.html

Croft, W. B., Metzler, D., & Strohman, T. (2009). *Search engines: Information retrieval in practice*. Addison-Wesley.

Dublin Core Metadata Initiative. (2020). *DCMI metadata terms*. https://www.dublincore.org/specifications/dublin-core/dcmi-terms/

Elmasri, R., & Navathe, S. B. (2015). *Fundamentals of database systems* (7.ª ed.). Pearson.

Floridi, L., & Cowls, J. (2019). A unified framework of five principles for AI in society. *Harvard Data Science Review*, *1*(1). https://doi.org/10.1162/99608f92.8cd550d1

Fox, E. A., & Gonçalves, M. A. (2007). *Building quality digital libraries: Macro-level architectures and methodologies*. Morgan & Claypool Publishers. https://doi.org/10.2200/S00059ED1V01Y200611DLM006

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). *Design patterns: Elements of reusable object-oriented software*. Addison-Wesley.

Gonçalves, M. A., Fox, E. A., Watson, L. T., & Kipp, N. A. (2004). Streams, structures, spaces, scenarios, societies (5S): A formal digital library framework. *ACM Transactions on Information Systems*, *22*(2), 270–312. https://doi.org/10.1145/984321.984325

Haak, L. L., Fenner, M., Paglione, L., Pentz, E., & Ratner, H. (2012). ORCID: A system to uniquely identify researchers. *Learned Publishing*, *25*(4), 259–264. https://doi.org/10.1087/20120404

Harold, E. R., & Means, W. S. (2004). *XML in a nutshell* (3.ª ed.). O'Reilly Media.

Hetland, M. L. (2017). *Beginning Python: From novice to professional* (3.ª ed.). Apress. https://doi.org/10.1007/978-1-4842-2385-7

International ISBN Agency. (2017). *ISBN users' manual* (7.ª ed.). https://www.isbn-international.org/content/isbn-users-manual

International Organization for Standardization. (2016). *Document management — Electronic document file format for long-term preservation — Part 1: Use of PDF 1.4 (PDF/A-1)* (ISO Standard No. 19005-1:2016). https://www.iso.org/standard/69389.html

International Organization for Standardization. (2017). *Information and documentation — The Dublin Core metadata element set* (ISO Standard No. 15836:2017). https://www.iso.org/standard/71339.html

Lagoze, C., & Van de Sompel, H. (2001). The Open Archives Initiative: Building a low-barrier interoperability framework. Em *Proceedings of the 1st ACM/IEEE-CS Joint Conference on Digital Libraries* (pp. 54–62). ACM. https://doi.org/10.1145/379437.379449

Lesk, M. (2004). *Understanding digital libraries* (2.ª ed.). Morgan Kaufmann / Elsevier. https://doi.org/10.1016/B978-1-55860-924-2.X5001-3

Lutz, M. (2013). *Learning Python* (5.ª ed.). O'Reilly Media.

Lynch, C. (2000). Digital collections, digital libraries and the digitization of cultural heritage. *First Monday*, *5*(5). https://doi.org/10.5210/fm.v5i5.749

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. Cambridge University Press. https://doi.org/10.1017/CBO9780511809079

O'Neil, C. (2016). *Weapons of math destruction: How big data increases inequality and threatens democracy*. Crown Publishing.

Paskin, N. (2010). Digital Object Identifier (DOI®) System. Em M. J. Bates & M. N. Maack (Eds.), *Encyclopedia of library and information sciences* (3.ª ed., pp. 1586–1592). CRC Press. https://doi.org/10.1081/E-ELIS3-120044418

Python Software Foundation. (2026). *json — JSON encoder and decoder* (Python 3.12.3 Documentation). https://docs.python.org/3/library/json.html

Ricci, F., Rokach, L., & Shapira, B. (Eds.). (2015). *Recommender systems handbook* (2.ª ed.). Springer. https://doi.org/10.1007/978-1-4899-7637-6

Salton, G., & McGill, M. J. (1983). *Introduction to modern information retrieval*. McGraw-Hill.

Suber, P. (2012). *Open access*. MIT Press. https://mitpress.mit.edu/9780262517638/open-access/

Tenopir, C., Volentine, R., & King, D. W. (2010). Managing digital collections: Usability and barriers in higher education. *Journal of Documentation*, *66*(2), 205–228. https://doi.org/10.1108/00220411011023624

World Wide Web Consortium. (2023). *EPUB 3.3 specification* (W3C Recommendation). https://www.w3.org/TR/epub-33/

Witten, I. H., Bainbridge, D., & Nichols, D. M. (2009). *How to build a digital library* (2.ª ed.). Morgan Kaufmann / Elsevier. https://www.elsevier.com/books/how-to-build-a-digital-library/witten/978-0-12-374857-7