Universidade Federal do Rio Grande do Sul
Faculdade de Biblioteconomia e Comunicação
Bacharelado em Arquivologia

Norman Schmidt Júnior

WORKFLOW DE MICROSSERVIÇOS DE PRESERVAÇÃO DIGITAL PARA OBJETOS DIGITALIZADOS DE ACERVOS EM INSTITUIÇÕES DE MEMÓRIA DE PEQUENO PORTE

Porto Alegre
2026

Norman Schmidt Júnior

WORKFLOW DE MICROSSERVIÇOS DE PRESERVAÇÃO DIGITAL PARA OBJETOS DIGITALIZADOS DE ACERVOS EM INSTITUIÇÕES DE MEMÓRIA DE PEQUENO PORTE

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de bacharel em Arquivologia da Faculdade de Biblioteconomia e Comunicação da Universidade Federal do Rio Grande do Sul.
Orientador: Prof. Dr. Rafael Port da Rocha

Porto Alegre
2026

CIP - Catalogação na Publicação

Schmidt Júnior, Norman
Workflow de microsserviços de preservação digital para objetos digitalizados de acervos em instituições de memória de pequeno porte / Norman Schmidt Júnior. -- 2026.
145 f.
Orientador: Rafael Port da Rocha.

Trabalho de conclusão de curso (Graduação) -- Universidade Federal do Rio Grande do Sul, Faculdade de Biblioteconomia e Comunicação, Curso de Arquivologia, Porto Alegre, BR-RS, 2026.

1. Preservação digital. 2. Instituição de memória de pequeno porte. 3. Microsserviços. 4. Software livre. I. Rocha, Rafael Port da, orient. II. Título.

Elaborada pelo Sistema de Geração Automática de Ficha Catalográfica da UFRGS com os dados fornecidos pelo(a) autor(a).

Norman Schmidt Júnior

WORKFLOW DE MICROSSERVIÇOS DE PRESERVAÇÃO DIGITAL PARA OBJETOS DIGITALIZADOS DE ACERVOS EM INSTITUIÇÕES DE MEMÓRIA DE PEQUENO PORTE

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de bacharel em Arquivologia da Faculdade de Biblioteconomia e Comunicação da Universidade Federal do Rio Grande do Sul.
Orientador: Prof. Dr. Rafael Port da Rocha

Aprovado em: Porto Alegre, _____ / _____ / _____

BANCA EXAMINADORA:

Prof. Dr. Rafael Port da Rocha (Orientador)
UFRGS

Prof. Jorge Eduardo Enriquez Vivar (Examinador)
UFRGS

Bel. Esp. Yuri Victorino Inacio da Silva (Examinador)
NUMIS Museu Numismático

à Isabela,
à Doris,
à Aline
e ao Próspero.

RESUMO

A literatura e as ferramentas de software pertinentes à preservação digital geralmente pressupõem a disponibilidade de equipes especializadas, infraestrutura de TI avançada e um alto nível de maturidade organizacional. Tais condições estão frequentemente ausentes em parte significativa das instituições de memória brasileiras, que custodiam patrimônio relevante operando com reduzidos recursos humanos, orçamentários e de tecnologia da informação. Diante desse cenário, este trabalho propõe um workflow de microsserviços essenciais de preservação digital, apoiado em software livre e exequível em equipamentos de TI de uso comum, voltado para essas instituições de memória de pequeno porte. A pesquisa, de natureza aplicada e caráter tecnológico, partiu da caracterização de aspectos da instituição de pequeno porte para a análise de oito modelos conceituais de curadoria e preservação digital de referência, onde foram identificados nove microsserviços essenciais, a partir de critérios de recorrência, escopo e governança. Para cada um foram selecionados aplicativos ou bibliotecas de software livre com base em critérios aqui definidos. Uma arquitetura de workflow foi desenhada para organizar a execução dos microsserviços em nove etapas de preservação e uma etapa periódica de verificação, com pontos de decisão e estratégias de falha, com a lógica de cada microsserviço implementada em scripts Python, e um script orquestrador para encapsular e controlar todas as etapas. O workflow foi submetido a uma avaliação funcional com cinco lotes de objetos digitais, compostos por itens do Format-Corpus da Open Preservation Foundation e por itens de controle, concebidos para exercitar o fluxo nominal e os pontos de decisão em casos de falha. Os resultados demonstraram que o workflow é exequível e que seus produtos — pacotes AIP válidos em formato BagIt, com metadados técnicos e de preservação, cadeia ininterrupta de eventos PREMIS, representações derivadas e cópias replicadas com integridade verificada — aderem a requisitos de preservação dos modelos conceituais e documentam uma cadeia de custódia que sustenta a autenticidade, viabilizando a abordagem de parcimônia digital do workflow, pelo qual instituições de memória de pequeno porte podem realizar ações de preservação digital com integridade, autenticidade e rastreabilidade verificáveis.

Palavras-chave: Preservação digital; Instituição de memória de pequeno porte; Microsserviços; Software livre.

ABSTRACT

Literature and software tools pertaining to digital preservation typically assume the availability of specialized teams, advanced IT infrastructure, and a high level of organizational maturity. Such conditions are often absent in a significant portion of Brazilian memory institutions, which safeguard relevant heritage while operating with limited human, budgetary, and information technology resources. Given this scenario, this work proposes a workflow of essential digital preservation microservices, supported by open-source software and feasible on commonly used IT equipment, aimed at these small-scale memory institutions. The research, of an applied and technological nature, began with the characterization of aspects of small-scale institutions to analyze eight conceptual models of digital curation and preservation, where nine essential microservices were identified based on criteria of recurrence, scope, and governance. For each, open-source software applications or libraries were selected based on criteria defined herein. A workflow architecture was designed to organize the execution of microservices into nine preservation steps and a periodic verification step, with decision points and failure strategies. The logic of each microservice was implemented in Python scripts, and an orchestrator script to encapsulate and controll all steps. The workflow underwent a functional evaluation with five batches of digital objects, composed of items from the Open Preservation Foundation's Format-Corpus and control items, designed to test the nominal flow and decision points in case of failure. The results demonstrated that the workflow is feasible and that its products—valid AIP packages in BagIt format, with technical and preservation metadata, an unbroken chain of PREMIS events, derived representations, and replicated copies with verified integrity—adhere to the preservation requirements of the conceptual models and document a chain of custody that sustains authenticity. This enables the workflow's digital parsimony approach, allowing small memory institutions to perform digital preservation actions with verifiable integrity, authenticity, and traceability.

Keywords: Digital preservation; Small-scale memory institution; Microservices; open-source software.

LISTA DE FIGURAS

Figura 1 — Entidades funcionais do modelo OAIS 22
Figura 2 — Modelo de ciclo de vida de curadoria do DCC.....23
Figura 3 — Ontologia de ações de preservação no modelo Dappert-Farquhar 26
Figura 4 — Microsserviços de curadoria digital.....28
Figura 5 — Relações conceituais entre categorias de objetos digitais.....35
Figura 6 — Aplicabilidade de microsserviços no ciclo de vida de curadoria e preservação.....39
Figura 7 — Visão geral da arquitetura do workflow.....87
Figura 8 — Fluxograma das etapas E1 a E9.....95
Figura 9 — Fluxograma da etapa E10, de verificação periódica de integridade (MS5) 96

LISTA DE QUADROS

Quadro 1 — Percurso metodológico por objetivo 45
Quadro 2 — Recorrência dos microsserviços candidatos nos modelos conceituais . 59
Quadro 3 — Microsserviços de preservação digital para instituições de memória de pequeno porte 61
Quadro 4 — Critérios de seleção dos aplicativos 65
Quadro 5 — Candidatos para a varredura antivírus 67
Quadro 6 — Candidatos para identificação de formato 68
Quadro 7 — Candidatos para validação de conformidade de formato 70
Quadro 8 — Candidatos para extração de metadados técnicos 71
Quadro 9 — Candidatos para verificação de integridade 74
Quadro 10 — Candidatos para empacotamento do objeto digital 75
Quadro 11 — Candidatos para geração registro de eventos de preservação 78
Quadro 12 — Candidatos para armazenamento com replicação 79
Quadro 13 — Candidatos para geração de derivadas 82
Quadro 14 — Síntese dos aplicativos selecionados por microsserviço 84
Quadro 15 — Etapas do workflow de microsserviços essenciais de preservação digital 97
Quadro 16 — Padrão de chamada do script orquestrador 100
Quadro 17 — Padrão de diretórios criados pelo microsserviço de empacotamento (MS6) 101
Quadro 18 — Scripts Python para o workflow de microsserviços de preservação digital 109
Quadro 19 — Ambiente de avaliação, máquina virtual e versões das ferramentas 112
Quadro 20 — Critérios de avaliação do workflow 113
Quadro 21 — Lotes de objetos para avaliação 115
Quadro 22 — Exemplo de comando para chamada de teste das etapas E1-E9 do workflow 116
Quadro 23 — Arquivo de configuração destinos.json, com destinos de replicação 116
Quadro 24 — Exemplo de comando para chamada de teste da etapa 10 do workflow 117
Quadro 25 — Evidência: árvore do AIP (resumida) 118
Quadro 26 — Evidência: validação BagIt (pós replicação) 119

Quadro 27 — Evidência: cadeia de eventos PREMIS, do log do orquestrador 119
Quadro 28 — Evidência: fixidez da matriz e relação matriz-derivada no PREMIS .. 120
Quadro 29 — Evidência: eventos de replicação, do log de replicação 121
Quadro 30 — Evidência: trecho do log do orquestrador (E4, E5 e E7) 122
Quadro 31 — Evidência: objeto PREMIS de arquivo não identificado (trecho): 123
Quadro 32 — Evidência: parecer JHOVE do modulo TIFF-hul de não conformidade (trecho) 123
Quadro 33 — Evidência: evento PREMIS de migração com resultado failure (trecho) 124
Quadro 34 — Evidência: notas de resultado dos eventos de E4, E5 e E7 (PREMIS) 125
Quadro 35 — Evidência: log de saída do ClamAV (E2) 126
Quadro 36 — Evidência: trecho do log do orquestrador (E2) 126
Quadro 37 — Evidência: evento PREMIS de rejeição (trecho) 126
Quadro 38 — Evidência: trecho do log do orquestrador (E2 aprovada, E3 reprovada) 128
Quadro 39 — Evidência: evento PREMIS de falha em verificação de fixidez (trecho) 128
Quadro 40 — Evidência: resultado por destino (eventos no log de replicação) 129
Quadro 41 — Evidência: trecho do log do orquestrador (E9) 130
Quadro 42 — Evidência: evento PREMIS replication com resultado failure (trecho) 130
Quadro 43 — Evidência: log do orquestrador da verificação (primeira execução) .. 131
Quadro 44 — Evidência: relatório CSV de integridade (segunda execução) 132
Quadro 45 — Evidência: evento PREMIS de falha de verificação de fixidez (trecho) 133
Quadro 46 — Síntese do atendimento aos critérios de avaliação 134

SUMÁRIO

1 INTRODUÇÃO – TEMA E PROBLEMATIZAÇÃO 14
1.1 JUSTIFICATIVA 15
1.2 OBJETIVOS 17
1.2.1 Objetivo Geral 17
1.2.2 Objetivos específicos 17
2 REFERENCIAL TEÓRICO 18
2.1 PRESERVAÇÃO DIGITAL 18
2.2 CURADORIA DIGITAL 19
2.3 MODELOS CONCEITUAIS DE PRESERVAÇÃO E CURADORIA DIGITAL 20
2.3.1 O Modelo OAIS para Sistema de Arquivamento de Informação 20
2.3.2 O modelo DCC e o ciclo de vida da Curadoria Digital 22
2.3.3 Planets 24
2.3.4 O modelo Dappert-Farquhar 25
2.3.5 O modelo UC3 (California Digital Library) 26
2.3.6 O modelo Pavão-Caregnato-Rocha 28
2.3.7 InterPARES 29
2.3.8 NDSA Levels of Digital Preservation 30
2.4 METADADOS DE PRESERVAÇÃO DIGITAL 31
2.4.1 Categorias de metadados 32
2.5 OBJETO DIGITAL 33
2.5.1 Representações do Objeto Digital 35
2.5.2 Propriedades Significativas 37
2.5.3 Empacotamento do Objeto Digital 37
2.6 MICROSSERVIÇOS DE PRESERVAÇÃO DIGITAL 39
3 METODOLOGIA 41
3.1 CARACTERIZAÇÃO DO CENÁRIO INSTITUCIONAL: A INSTITUIÇÃO DE MEMÓRIA DE PEQUENO PORTE 41
3.2 PERCURSO METODOLÓGICO 44
4 MICROSSERVIÇOS DE PRESERVAÇÃO DIGITAL PARA INSTITUIÇÕES DE MEMÓRIA DE PEQUENO PORTE 50

4.1 MICROSSERVIÇOS DE PRESERVAÇÃO DIGITAL NOS MODELOS CONCEITUAIS 51
4.1.1 Microsserviços no modelo OAIS 52
4.1.2 Microsserviços no modelo DCC de ciclo de vida da curadoria digital. 53
4.1.3 Microsserviços no projeto PLANETS 54
4.1.4 Microsserviços no modelo Dappert-Farquhar 55
4.1.5 Microsserviços no modelo UC3 55
4.1.6 Microsserviços no modelo Pavão-Caregnato-Rocha 56
4.1.7 Microsserviços no projeto InterPARES 58
4.1.8 Microsserviços no framework NDSA Levels of Digital Preservation ... 58
4.2 MICROSSERVIÇOS ESSENCIAIS PARA UMA INSTITUIÇÃO DE MEMÓRIA DE PEQUENO PORTE 59
5 APLICATIVOS DE SOFTWARE LIVRE QUE REALIZAM MICROSSERVIÇOS DE PRESERVAÇÃO DIGITAL 65
5.1 MICROSSERVIÇO DE VALIDAÇÃO DE ADMISSÃO (AV) 66
5.2 MICROSSERVIÇO DE IDENTIFICAÇÃO DE FORMATO 68
5.3 MICROSSERVIÇO DE VALIDAÇÃO DE CONFORMIDADE DE FORMATO 69
5.4 MICROSSERVIÇO DE EXTRAÇÃO DE METADADOS TÉCNICOS 71
5.5 MICROSSERVIÇO DE VERIFICAÇÃO DE INTEGRIDADE - FIXIDEZ ... 73
5.6 MICROSSERVIÇO DE EMPACOTAMENTO DO OBJETO DIGITAL (SIP/AIP) 75
5.7 MICROSSERVIÇO DE REGISTRO DE EVENTOS DE PRESERVAÇÃO 77
5.8 MICROSSERVIÇO DE ARMAZENAMENTO COM REPLICAÇÃO 79
5.9 MICROSSERVIÇO DE GERAÇÃO DE REPRESENTAÇÕES DERIVADAS 80
5.10 SINTESE DOS APLICATIVOS SELECIONADOS 83
6 WORKFLOW PARA MICROSSERVIÇOS ESSENCIAIS DE PRESERVAÇÃO DIGITAL 86
6.1 PRINCIPIOS DE ORGANIZAÇÃO DO WORKFLOW 86
6.2 ESTRUTURA DO WORKFLOW 87
6.3 FASES E ETAPAS 88
6.3.1 Fase de recepção, triagem e validação, etapas E1 a E3 88

6.3.1.1 Etapa E1: Recepção e preparação do material 89
6.3.1.2 Etapa E2: Validação de admissão (MS1) 89
6.3.1.3 Etapa E3: Constituição ou validação do SIP (MS6) 89
6.3.2 Fase de caracterização e empacotamento, etapas E4 a E8 90
6.3.2.1 Etapa E4: Identificação de formato (MS2) 90
6.3.2.2 Etapa E5: Validação de conformidade de formato (MS3) 91
6.3.2.3 Etapa E6: Extração de metadados técnicos (MS4) 91
6.3.2.4 Etapa E7: Geração de representações derivadas (MS9) 92
6.3.2.5 Etapa E8: Empacotamento do AIP (MS6) 92
6.3.3 Fase de armazenamento, etapa E9 93
6.3.3.1 Etapa E9: Armazenamento e replicação (MS8) 93
6.3.4 Fase de Manutenção, etapa E10 93
6.3.4.1 Etapa E10: Verificação periódica de integridade (MS5) 94
6.4 PONTOS DE DECISÃO EM FALHAS 94
6.5 QUADRO/SINTESE DAS FASES/ETAPAS 97
7 SCRIPTS PARA REALIZAR MICROSSERVIÇOS NO WORKFLOW 100
7.1 ESTRUTURA / CONVENÇOES 100
7.2 ORQUESTRADOR.PY 101
7.3 ADMISSAO.PY 102
7.4 EMPACOTAMENTO.PY 103
7.5 IDENTIFICACAO.PY 104
7.6 CONFORMIDADE.PY 104
7.7 CARACTERIZACAO.PY 105
7.8 DERIVADAS.PY 105
7.9 EVENTOS.PY 106
7.10 REPLICACAO.PY 107
7.11 VERIFICACAO.PY 107
7.12 QUADRO SINTESE DOS SCRIPTS PARA REALIZAÇÃO DE MICROSSERVIÇOS 108
8 AVALIAÇÃO 111
8.1 NATUREZA DA AVALIAÇÃO 111
8.2 AMBIENTE DE AVALIAÇÃO 112
8.3 CRITÉRIOS DE AVALIAÇÃO 113
8.4 LOTES DE OBJETOS DE AVALIAÇÃO 114

8.5 EXECUÇÃO 116
8.6 RESULTADOS 117
8.6.1 Lote L1 — Lote íntegro 118
8.6.2 Lote L2 — Lote misto 122
8.6.3 Lote L3 — Vírus na admissão (ponto de decisão E2) 125
8.6.4 Lote L4 — Pacote corrompido (ponto de decisão E3) 127
8.6.5 Lote L5 — Destino indisponível (ponto de decisão E9) 129
8.6.6 Verificação periódica de integridade (Etapa E10) 131
8.6.7 Síntese dos resultados 133
8.7 ANÁLISE DOS RESULTADOS 134
9 CONCLUSÃO 136
REFERÊNCIAS 140
APÊNDICE A — REPOSITÓRIO DE SCRIPTS DO WORKFLOW 145

1 INTRODUÇÃO – TEMA E PROBLEMATIZAÇÃO

A implantação da preservação digital em instituições é uma tarefa complexa, pois envolve a integração de conceitos de áreas distintas entre as Ciências da Informação e as Ciências da Computação. Essa implantação fica ainda mais complexa quando pretendemos usar ferramentas baseadas em software para automatizar tarefas repetitivas, ou para converter ou adequar automaticamente os dados a regras definidas para sua preservação, pois, para isso, devemos compreender e compatibilizar os conceitos de preservação que embasam ou que podem ser contemplados pelo software com os conceitos de preservação assumidos pela instituição que deseja efetivamente praticar a preservação digital.

Para documentar e fornecer subsídios para ações de Preservação Digital, foram desenvolvidos modelos conceituais como o Open Archival Information System (OAIS) e o Projeto InterPARES (International Research on Permanent Authentic Records in Electronic Systems).

Esses modelos conceituais acabaram por mostrar que a Preservação Digital poderia observar e contemplar mais do que simples preservação, catalogação e cruzamento de referências entre objetos digitais estáticos. O volume de informações produzido nativamente em forma digital pela sociedade moderna e a facilidade cada vez maior de acesso a processos de digitalização de documentos não digitais, tornou imperativo que houvesse meios de agregar valor informacional, organizacional e estrutural às próprias informações contidas nesses documentos digitais preservados.

A conceituação da Curadoria Digital levou à concepção de modelos conceituais de Preservação Digital que contemplavam também aspectos da Curadoria Digital, dentre eles o Projeto PLANETS (Farquhar; Hockx-Yu, 2007), o modelo Dappert-Farquhar (Dappert; Farquhar, 2009), o modelo de ciclo de vida de curadoria digital desenvolvido pelo Digital Curation Centre (Higgins, 2008), os fundamentos de curadoria digital desenvolvidos pelo University of California Curation Center (UC3) (California Digital Library, 2010) e o modelo Pavão, Caregnato e Rocha (Pavão; Caregnato; Rocha, 2016).

Esses modelos, ao especificarem atividades e representações de informação para a curadoria digital, exigem a definição de uma grande quantidade de atividades computacionais, incluindo microsserviços de curadoria digital que realizam avaliações, testes e validações dos objetos digitais ao longo de todos os seus ciclos de vida. Com

isso, começaram a surgir ferramentas de software com o objetivo era de reunir diversas ferramentas computacionais para o processamento dessas atividades individuais em conjuntos, ou sequências cuja lógica se atrela aos modelos conceituais.

Atualmente existem ferramentas que automatizam e validam processos de curadoria digital, como o Preservica e o Libnova Libsafe, ou projetos de código aberto como o Archivematica, o RODA Community Edition e o Samvera. Entretanto, essas ferramentas são complexas e sofisticadas, sendo adequadas às instituições de grande porte, que dispõem de infraestruturas de TI amplas e especializadas, voltadas a atender vários sistemas administrativos e de negócio. Isso acaba excluindo as instituições de pequeno porte, isto é, aquelas em que a infraestrutura de TI se restringe a prover redes locais, internet e alguns serviços disponibilizados em servidores de pequeno porte, como servidores Linux.

Muitos arquivos, museus e bibliotecas estaduais e municipais enquadram-se neste perfil. São custodiadoras de acervos que demandam preservação digital de longo prazo, mas operam com equipes não especializadas, orçamentos reduzidos e infraestrutura computacional limitada, em condições muito distantes daquelas exigidas para a implementação bem-sucedida das ferramentas de mercado atualmente disponíveis. Tanto as soluções proprietárias quanto alternativas de código aberto podem demandar capacidade institucional, técnica e de infraestrutura que essas instituições não possuem. A literatura especializada normalmente documenta cenários institucionais com maturidade tecnológica elevada, oferecendo pouca orientação prática para contextos de escassez de recursos.

Diante desse quadro, este trabalho busca responder à seguinte pergunta de pesquisa: Como estruturar um ambiente computacional composto por microsserviços essenciais de preservação digital, apoiado em software livre, que seja exequível para instituições de memória de pequeno porte?

1.1 JUSTIFICATIVA

As barreiras financeiras ou estruturais são particularmente importantes para uma parte significativa de instituições custodiadoras do patrimônio documental brasileiro. São elas: arquivos comunitários, centros documentais universitários, acervos de associações civis, arquivos pessoais e outras instituições de memória que não dispõem de grandes recursos de infraestrutura e pessoal. Durante a graduação o

autor visitou diversas instituições de portes distintos e observou que, se de um lado encontram-se instituições governamentais com equipes especializadas, infraestrutura computacional dedicada e políticas formais de preservação; no outro existem muitas instituições que custodiam acervos documentais de todos os portes, mas que funcionam com equipes e orçamentos reduzidos, equipamentos de uso geral ou compartilhado e que não dispõem, de forma satisfatória, de profissionais da área de Ciência da Informação em seus quadros funcionais. O Relatório Técnico sobre Censo de Arquivos Públicos Instituídos no Brasil (CONARQ, 2020a) evidencia esse desequilíbrio ao demonstrar que uma parte expressiva dos municípios brasileiros não dispõe de arquivo público estruturado. Para essas instituições, as ferramentas de curadoria digital disponíveis no mercado não oferecem uma resposta acessível: os sistemas comerciais implicam em altos custos de licenciamento, infraestrutura e pessoal especializado, e os projetos de código aberto também requerem capacidade técnica e infraestrutura que normalmente ultrapassam as possibilidades dessas instituições.

Identificar microsserviços essenciais de preservação digital e instrumentalizá-los com ferramentas de software livre executáveis em equipamentos de uso geral é, portanto, uma contribuição com impacto direto sobre esse segmento institucional, ao qual os modelos e as ferramentas existentes não atendem adequadamente.

Sob o ponto de vista científico, o presente trabalho justifica-se pela procura e produção de novas estratégias para implantação de Preservação Digital em uma abordagem focada na parcimônia digital, identificando os microsserviços de preservação digital que são essenciais a partir da análise de funções de curadoria propostas nos modelos conceituais de curadoria e preservação digital, e instrumentalizando-os com ferramentas de código aberto e software livre, de baixo consumo de recursos.

Sob o ponto de vista pessoal, o autor trabalha profissionalmente na área de software há mais de vinte e cinco anos, e ficou extremamente interessado em aplicar sua grande experiência em desenvolvimento de software e práticas de uso de soluções de software baseadas em software livre para criar novas informações e documentação que possam contribuir para sua área de interesse de estudo, a Arquivologia e a Preservação Digital.

1.2 OBJETIVOS

Dividiram-se os objetivos em um objetivo geral e cinco objetivos específicos, definidos a seguir.

1.2.1 Objetivo Geral

Propor e elaborar um ambiente computacional composto por microsserviços essenciais de preservação digital, voltado a instituições de memória de pequeno porte.

1.2.2 Objetivos específicos

a) identificar os microsserviços essenciais de preservação digital para instituições de memória de pequeno porte;
b) analisar aplicativos e bibliotecas de software livre, disponíveis na plataforma Linux, que apoiam ou realizam a implementação de microsserviços essenciais de preservação digital;
c) propor uma arquitetura baseada em um workflow, para a execução dos microsserviços essenciais de preservação digital na plataforma Linux;
d) implementar, por meio de scripts Python, a orquestração do workflow e o encapsulamento dos microsserviços, realizados por aplicativos e bibliotecas disponíveis no ambiente Linux;
e) avaliar a execução do workflow e seus resultados, sobre elementos de um acervo de objetos digitais obtidos no corpus do Projeto Format-Corpus da Open Preservation Foundation (OPF).

2 REFERENCIAL TEÓRICO

2.1 PRESERVAÇÃO DIGITAL

O Conselho Nacional de Arquivos, em seu Glossário de Documentos Arquivísticos Digitais (CONARQ, 2020b), define preservação digital como o conjunto de ações gerenciais e técnicas necessárias para superar as mudanças tecnológicas e a fragilidade dos suportes, de modo a garantir acesso e interpretação dos documentos digitais pelo tempo necessário. Definição equivalente encontra-se no modelo de referência OAIS (CCSDS, 2012), que caracteriza a preservação digital como o ato de manter, ao longo do tempo, a acessibilidade e a inteligibilidade da informação digital diante das mudanças do ambiente tecnológico e de mudanças na comunidade de usuários.

Os problemas para a preservação digital já eram discutidos desde a década de 1970, porém foi somente em 1994 que, nos Estados Unidos, membros oriundos do Research Libraries Group (RLG) e da Commission on Preservation and Access (CPA) formaram um grupo de trabalho que produziu um relatório considerado fundamental no tema: “Preserving digital information: Report of the task force on archiving of digital information” (Preservando informações digitais: Relatório da força-tarefa em arquivamento de informações digitais, em tradução própria). Nesse relatório foi proposta a criação de um sistema nacional de arquivos digitais que ofereceria armazenamento e acesso a informações digitais. Além disso, foram introduzidos conceitos de repositórios digitais confiáveis, e identificadas várias características de integridade de dados digitais que depois seriam incorporadas ao modelo OAIS. O relatório também discorreu sobre a importância da ação de migração de dados, considerando-a fundamental para a Preservação Digital. (Waters; Garrett, 1996)

O conceito de Repositório Digital Confiável foi desenvolvido em publicação produzida em 2000 pelo RLG em associação com o Online Computer Library Center (OCLC). Nessa publicação, um Repositório Digital Confiável é definido como um repositório cuja missão é prover acesso confiável de longo prazo a recursos digitais para sua comunidade de usuários, no presente e no futuro (Research Libraries Group; OCLC, 2002). No Brasil, esse conceito foi adotado pelo CONARQ na Resolução nº 43/2015, que estabeleceu os requisitos para a implementação de Repositórios Arquivísticos Digitais Confiáveis (RDC-Arq) nas instituições públicas do país (CONARQ, 2015).

A partir desses marcos fundamentais originaram-se diversos modelos conceituais voltados a formalizar requisitos, atividades e arquiteturas necessárias para a preservação digital. Esses modelos, apresentados na seção 2.3, constituem a base teórica sobre a qual se sustenta a identificação e a organização dos microsserviços essenciais propostos neste trabalho para execução em um workflow simplificado.

2.2 CURADORIA DIGITAL

Uma das conceituações pioneiras de curadoria digital foi publicada em 2003, em uma Circular do Joint Information Systems Committee (JISC), uma entidade britânica sem fins lucrativos da área de suporte educacional. Nessa publicação foram delineados vários conceitos, escopo e dificuldades da curadoria digital:

> Curadoria Digital define-se aqui como as ações necessárias para a manutenção de dados de pesquisa e outros materiais digitais através de seus ciclos de vida e através do tempo, para os usuários atuais e futuros. Implícitos nesta definição estão os processos de arquivamento e preservação digital, e também todos os processos para criação e adição de valor aos dados originais para geração de novas fontes de informação e conhecimento. A Curadoria e a preservação de longo prazo de recursos digitais se tornarão cada vez mais importantes em uma ampla gama de atividades de pesquisa e educação (Joint Information Systems Committee, 2003, tradução própria).

A conceituação da curadoria digital impulsionou o desenvolvimento de modelos conceituais que passaram a contemplar não apenas a preservação dos objetos digitais em sentido estrito, mas também os processos de caracterização, planejamento, transformação e acesso que garantem sua usabilidade ao longo do tempo. A curadoria digital constitui, assim, o domínio mais amplo, que articula a preservação, o acesso e a agregação de valor ao longo do ciclo de vida do objeto digital. Desse domínio, o presente trabalho recorta especificamente as funções de preservação digital que atuam sobre as matrizes dos objetos — ingestão, preservação, transformação e armazenamento — mantendo fora do escopo as funções de seleção, acesso, uso e monitoramento, conforme delimitado adiante na Seção 4. Esses modelos são apresentados na seção seguinte.

2.3 MODELOS CONCEITUAIS DE PRESERVAÇÃO E CURADORIA DIGITAL

A preservação e a curadoria digital são sustentadas por modelos conceituais desenvolvidos ao longo das últimas três décadas a partir de iniciativas acadêmicas, governamentais e de organizações nacionais e internacionais. Esses modelos têm naturezas distintas: alguns definem arquiteturas de sistemas e vocabulários de referência, como o modelo OAIS; outros propõem ciclos de vida e frameworks de atividades, como o DCC e o UC3; outros ainda formalizam requisitos de preservação e curadoria em termos operacionalizáveis, como o PLANETS e o modelo Dappert-Farquhar; outros oferecem frameworks de avaliação de maturidade e orientação prática, como o InterPARES e o NDSA Levels of Digital Preservation; e outros como o Pavão-Caregnato-Rocha sintetizam diversos modelos em um só. Em conjunto, esses modelos constituem a base conceitual sobre a qual são identificados, justificados e organizados os microsserviços de preservação digital usados nas seções posteriores para composição do workflow simplificado que este trabalho propõe.

O modelo OAIS apresentado na subseção 2.3.1, foi publicado originalmente em 2002, ocupa posição de destaque histórico e conceitual em relação aos demais e é uma referência fundamental para todos os modelos posteriores. Muitos o adotam como ponto de partida ou integram seus conceitos explicitamente. As subseções seguintes apresentam os demais modelos na ordem de sua contribuição para a construção do referencial deste trabalho.

2.3.1 O Modelo OAIS para Sistema de Arquivamento de Informação

O modelo OAIS (Open Archival Information System) representa um framework conceitual onde são definidos elementos de vocabulário, estruturas funcionais e os requisitos para a preservação de longo prazo de objetos digitais em repositórios digitais (CCSDS, 2012).

Esse modelo especifica o ambiente em que o repositório digital está inserido, o modelo da informação a ser armazenada, assim como as principais funções a serem realizadas por um repositório, denominadas de componentes funcionais. O modelo formaliza também a importância da definição da Comunidade Designada, o conjunto

de usuários que compõe o público-alvo consumidor dos objetos desse arquivo, o qual detém o conhecimento que orientará as estratégias de preservação (Lavoie, 2014)

Em relação ao modelo de informação, o principal conceito é o de Pacote de Informação, que pode assumir três tipos dentro do ciclo de preservação digital:

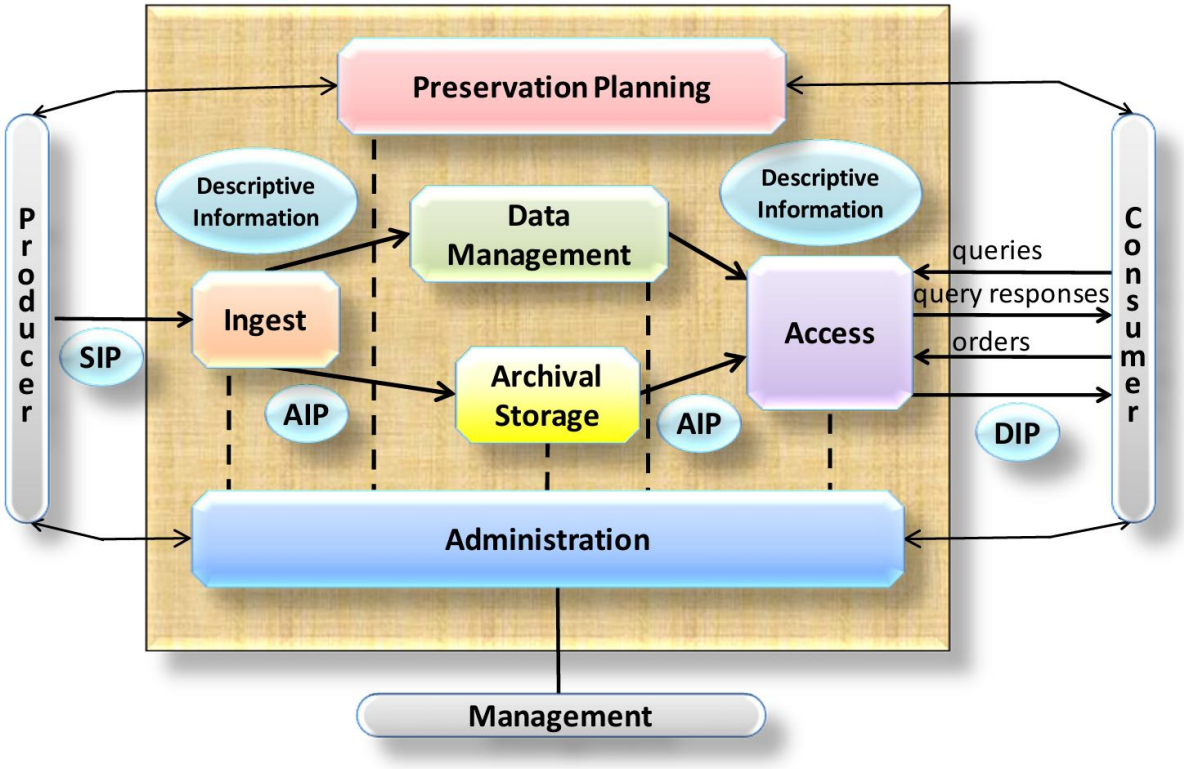
a) o SIP, ou Pacote de submissão, que é entregue pelo produtor no momento da ingestão dos objetos digitais;
b) o AIP, ou Pacote de Arquivamento, que representa a forma canônica de armazenamento, orientada para a garantia de autenticidade e integridade;
c) o DIP, ou Pacote de Disseminação, que é gerado para consumo final pelos usuários.

Funcionalmente, o modelo OAIS especifica seis entidades interdependentes (CCSDS, 2012; Thomaz; Soares, 2004):

a) a Ingestão, que recebe e valida os objetos;
b) o Armazenamento de Arquivo, onde é feita a guarda física e verificações periódicas de integridade;
c) o Gerenciamento de Dados, onde são administradas bases de metadados técnicos que descrevem os objetos;
d) a Administração, que aplica políticas institucionais e acordos com os produtores dos objetos;
e) o Planejamento de Preservação, que monitora e planeja estratégias para lidar com a obsolescência tecnológica através de migração de formatos ou emulação;
f) o Acesso, que implementa as interfaces para recuperação e disseminação dos objetos em formatos consumíveis.

A figura 1 mostra as relações entre as entidades funcionais do modelo, sua relação com o ambiente (produtor, consumidor e gestão) e com os pacotes de informação (SIP, AIP e DIP):

Figura 1 — Entidades funcionais do modelo OAIS



Fonte: Consultative Committee for Space Data Systems (2024).

No Brasil, o modelo OAIS teve influência explícita na Resolução nº 43/2015 do Conselho Nacional de Arquivos (CONARQ), que adotou seus princípios para a implementação de Repositórios Arquivísticos Digitais Confiáveis (RDC-Arq) nas instituições públicas do país (CONARQ, 2015).

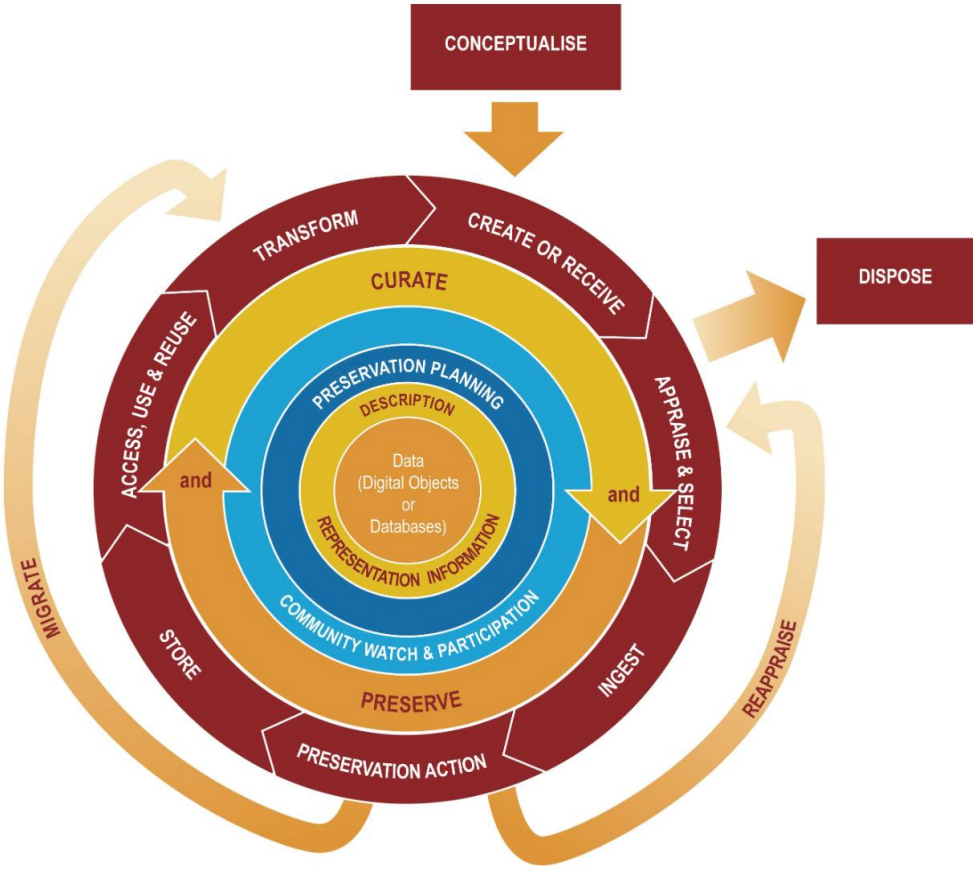
2.3.2 O modelo DCC e o ciclo de vida da Curadoria Digital

O modelo DCC de ciclo de vida da curadoria digital provê uma representação gráfica dos estágios necessários para o sucesso nas atividades de curadoria e preservação de dados digitais, desde sua concepção inicial e ao longo de um ciclo iterativo de curadoria (Higgins, 2008).

No centro do modelo estão os próprios dados, em torno dos quais as demais ações se organizam em camadas concêntricas, evidenciando que toda a curadoria existe em função da manutenção do objeto digital ao longo do tempo (Figura 2). A partir desse núcleo, o modelo distingue três conjuntos de ações:

a) ações ao longo de todo o ciclo de vida, que acompanham continuamente o objeto digital, sem um momento fixo de execução: a descrição e a informação de representação (atribuição de metadados e da informação que mantém o objeto compreensível e utilizável), o planejamento da preservação, o monitoramento da participação da comunidade e a ação de curadoria e preservação;
b) ações sequenciais, executadas em sequência na ordem do fluxo de curadoria: conceitualizar, criar e receber, avaliar e selecionar, ingerir, ação de preservação, armazenar, acessar, usar, reusar e transformar;
c) ações ocasionais, executadas ocasionalmente em situações específicas: descartar os objetos não selecionados, reavaliar objetos que falham na validação e migrar para outro formato para evitar a obsolescência tecnológica.

Figura 2 — Modelo de ciclo de vida de curadoria do DCC



Fonte: Higgins (2008, p. 137).

O modelo DCC foi desenvolvido para ser usado em planejamento das atividades dentro de projetos de pesquisa específicos, organizações ou consórcios,

para assegurar que todos os estágios necessários sejam executados na sequência correta. É importante notar que as atividades de Descrição, Planejamento de preservação, Observação da comunidade e Curadoria e Preservação devem ser consideradas em todos os estágios da atividade. (Higgins, 2008)

2.3.3 Planets

O projeto Planets, ou Preservation and Long-term Access through Networked Services, foi um projeto de desenvolvimento tecnológico subsidiado pela União Europeia com o objetivo de elaborar ferramentas e serviços de software práticos para ajudar a preservação digital de longo prazo de dados culturais e científicos em formato digital.

O projeto desenvolveu um ambiente integrado para desenvolvimento e avaliação de estratégias de preservação para dados culturais, focando nos requisitos de preservação de instituições repositórias de heranças culturais, como bibliotecas e arquivos nacionais. (Farquhar; Hockx-Yu, 2007).

Os mesmos autores organizam o modelo funcional do projeto em quatro funções de alto nível, que estruturam o tratamento dado aos objetos digitais ao longo da preservação:

a) observação (Preservation Watch), uma função que monitora o conteúdo, os usuários, a organização, o produtor e o ambiente tecnológico para definir requisitos de preservação, além de atualizar informações de representação e emitir alertas quando uma ação de preservação precisa ser executada;
b) planejamento da preservação (Preservation Planning), uma função onde são avaliados requisitos e é selecionada a solução de preservação mais adequada, além de solicitar o desenvolvimento de novas ações de preservação quando nenhuma escolha adequada estiver disponível;
c) ação de preservação (Preservation Action), uma função que executa as ações de preservação necessárias sobre os objetos digitais para assegurar o acesso continuado e desenvolve novas ferramentas para implementar novas ações, quando solicitadas pelo planejamento da preservação;

d) caracterização (Preservation Characterisation), uma função que dá suporte às atividades de ingestão e de ações de preservação, identificando formatos de arquivo, extraindo metadados e comparando as características dos objetos digitais antes e depois de uma ação de preservação.

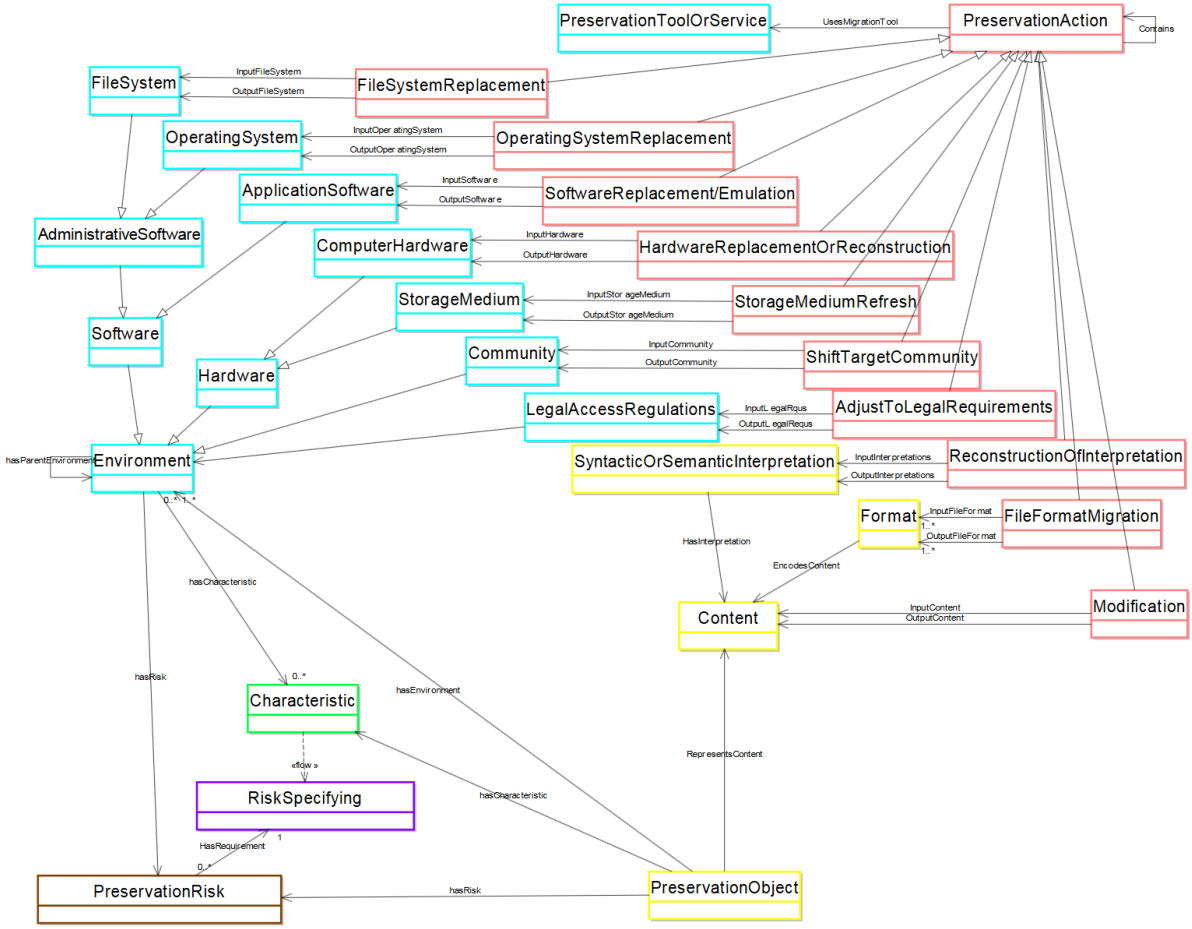
2.3.4 O modelo Dappert-Farquhar

O modelo Dappert-Farquhar introduz um modelo conceitual e um vocabulário orientado à Preservação de documentos. Ele mostra como o modelo e o vocabulário podem ser usados para especificar os requisitos de Preservação e Curadoria preferencialmente de forma automatizável e como esses requisitos podem então ser usados para promover a preservação e curadoria digital, através da caracterização, avaliação e monitoramento de ações, requisitos e riscos de preservação (Dappert; Farquhar, 2009).

O modelo conceitual apresenta uma representação simples, porém expressiva do domínio e escopo de um plano de preservação. O modelo e vocabulário podem ser disponibilizados através de ferramentas de software, e oferecem um bom ponto de partida para criação de modelos personalizados para instituições com características específicas, mesmo em casos de instituições que não exigem que as especificações de preservação sejam interpretáveis por máquina (Dappert; Farquhar, 2009).

O modelo especifica diversas ações de preservação e as relações que elas mantêm com o ambiente, com o objeto digital e com os riscos de preservação decorrentes de mudanças nesse ambiente. A Figura 3 exemplifica essas relações, representando as ações de preservação (em vermelho) e os elementos do ambiente (em azul) e do objeto digital (em amarelo) por elas afetados.

Figura 3 — Ontologia de ações de preservação no modelo Dappert-Farquhar



Fonte: Dappert e Farquhar (2009, p. 128).

Ele ainda provê formas de associar processos à uma ampla gama de entidades de cada instituição, coleção ou arquivo, até os níveis mais básicos dos componentes lógicos de objetos digitais, observando as propriedades organizacionais e técnicas da entidade e do acervo, e considerando todos os tipos de ações de preservação: De ações de software (migração, emulação, reparação de arquivos) a ações de hardware (substituição de unidades físicas de armazenamento, substituição de hardware de processamento, reparo de hardware) sem esquecer ações organizacionais (adaptação de processos a mudanças de legislação, adaptação a novos requisitos da comunidade de usuários) (Pavão; Caregnato; Rocha, 2016).

2.3.5 O modelo UC3 (California Digital Library)

O modelo UC3 foi criado pelo University of California Curation Center (UC3), um programa de curadoria digital da California Digital Library (CDL) dentro do sistema

da Universidade da Califórnia. As definições desse modelo foram reunidas no documento UC3 Curation Foundations (California Digital Library, 2010), onde foram estabelecidos os fundamentos conceituais adotados pelo programa de curadoria da CDL para o tratamento de objetos digitais para preservação.

Já na introdução do documento é feita a declaração de que a missão do modelo é a curadoria digital, e que esta é entendida como uma noção mais abrangente do que a preservação digital, apontando definições que justificam essa declaração, e concluindo que a curadoria representa a coordenação entre atividades de preservação e de acesso ao material preservado, e que embora frequentemente tratadas como díspares, essas atividades são na verdade complementares: A preservação tem por objetivo oferecer disponibilidade do conteúdo preservado ao longo do tempo, enquanto o acesso depende da preservação realizada até o ponto do tempo em que o acesso ocorre. Nessa perspectiva, a curadoria não se limita a preservar o conteúdo original tal como foi recebido em um dado momento, mas também busca enriquecer as informações agregando valor informacional ao conteúdo durante todo o ciclo de vida de curadoria e preservação (California Digital Library, 2010).

A principal contribuição do modelo UC3 para este trabalho, porém, está na arquitetura através da qual ele propõe operacionalizar sua missão. Em vez de um sistema integrado e monolítico, o UC3 estrutura um sistema de curadoria como um conjunto de microsserviços: pequenas unidades funcionais autônomas e especializadas, cada uma responsável por uma única ação bem delimitada sobre os objetos digitais ou sobre os metadados a eles associados, sendo combináveis entre si para compor fluxos de maior complexidade (Abrams; Kunze; Loy, 2010). Pela sua granularidade, independência e baixo acoplamento, esses serviços podem ser desenvolvidos, mantidos, substituídos e recombinados de forma flexível, permitindo que cada instituição organize fluxos de preservação de acordo com suas necessidades específicas, e adaptados à infraestrutura de que dispõe. A Figura 4 apresenta o conjunto de microsserviços que compõem essa arquitetura de curadoria proposta pelo modelo UC3.

Figura 4 — Microsserviços de curadoria digital

Mode	Focus	Layer / micro-service
Curation	*Value*	*Interoperation* • Annotation • Notification
	Service	*Application* • Transformation • Search • Index • Ingest
Preservation	*Context*	*Interpretation* • Characterization • Inventory
	State	*Protection* • Replication • Fixity • Storage • Identity

Fonte: Abrams; Kunze; Loy, 2010

Foi nesse cenário institucional que teve origem o paradigma de microsserviços aplicado à preservação digital, razão pela qual o modelo UC3 constitui uma das principais referências conceituais para a abordagem de microsserviços adotada neste trabalho. Essa flexibilidade arquitetural mostra-se especialmente vantajosa em instituições de memória de pequeno porte, nas quais os recursos tecnológicos e humanos são restritos.

2.3.6 O modelo Pavão-Caregnato-Rocha

O modelo Pavão-Caregnato-Rocha é um modelo conceitual sintetizado a partir de diversos outros modelos conceituais de Preservação Digital, de curadoria digital e de gestão de repositórios confiáveis e de gestão de metadados. O modelo foi compilado a partir de diversas especificações de múltiplos modelos e projetos influentes e bem conceituados na área de Preservação e Curadoria digital: Do modelo Dappert-Farquhar (Dappert; Farquhar, 2009), das definições de padrões de metadados para Preservação digital PREMIS (PREMIS Editorial Committee, 2015),

do modelo de ciclo de vida de curadoria digital DCC (Higgins, 2008), dos fundamentos de curadoria digital desenvolvidos pelo University of California Curation Center (UC3) (California Digital Library, 2010) e do modelo de referência para arquivos e repositórios digitais OAIS (CCSDS, 2012).

É um modelo que evidencia a importância da curadoria no processo de preservação digital, e contempla serviços (planejamento, ingestão, armazenamento), políticas e estratégias de preservação, a composição do plano de preservação (dividido em ação, requisitos e riscos de preservação), os objetos de preservação, o ambiente (software, hardware, comunidade e produtor), eventos e agentes (Pavão; Caregnato; Rocha, 2016).

2.3.7 InterPARES

O InterPARES (International Research on Permanent Authentic Records in Electronic Systems) é uma iniciativa de pesquisa acadêmica colaborativa internacional, coordenada pela Universidade da Colúmbia Britânica (UBC), voltada à investigação dos requisitos para a preservação de longo prazo de registros digitais autênticos. O projeto tem suas raízes no Projeto UBC, desenvolvido entre 1994 e 1997, cujo objetivo era identificar e estabelecer os requisitos para a produção, manutenção e preservação de documentos eletrônicos com valor probatório (Duranti, 2001). A partir desse núcleo inicial, o InterPARES desenvolveu-se em quatro fases sucessivas, cada uma ampliando o escopo e o tipo de registros investigados.

A primeira fase, InterPARES 1 (1999–2001), investigou requisitos para a preservação de autenticidade de registros digitais, definindo os critérios de autenticidade como um conjunto de características que um registro deve manter ao longo do tempo para ser considerado genuíno e não adulterado, e as condições necessárias para a avaliação desses critérios. A segunda fase, InterPARES 2 (2002–2007), expandiu o escopo para registros produzidos em ambientes interativos e dinâmicos, incluindo registros de artes, ciências e governo eletrônico (Duranti; Preston, 2008). A terceira fase, InterPARES 3 (2007–2012), expandiu o escopo do projeto para incluir estudos de caso em diferentes contextos institucionais e produziu documentos de orientação prática voltados à implementação de estratégias de preservação (InterPARES 3 Project Team, 2012). A quarta fase, InterPARES Trust

(2013–2019), investigou a confiabilidade de registros digitais armazenados em ambientes de nuvem (Duranti; Rogers, 2019).

Do ponto de vista conceitual, a contribuição mais relevante do InterPARES para este trabalho reside na formalização da importância do conceito de autenticidade como um requisito essencial da preservação digital. Para o InterPARES, um registro digital é autêntico quando mantém sua identidade, como o conjunto de características que o distinguem de qualquer outro registro, quando mantém sua integridade, na condição de estar completo e inalterado, e quando é possível demonstrar que essas propriedades foram mantidas ao longo de sua cadeia de custódia (Duranti, 2005).

Essas definições sustentam diretamente dois dos microsserviços essenciais identificados na Seção 4: a verificação de integridade e o registro de eventos de preservação. Este último possibilita implementar o conceito da rastreabilidade da cadeia de custódia ininterrupta do InterPARES, segundo o qual cada transferência, transformação ou ação sobre um registro deve ser documentada de forma a não deixar lacunas na história do objeto desde sua criação até o presente.

2.3.8 NDSA Levels of Digital Preservation

Os NDSA Levels of Digital Preservation (Níveis de Preservação Digital da NDSA) são um framework de avaliação e orientação desenvolvido pela National Digital Stewardship Alliance (NDSA), uma coalizão de organizações norte-americanas voltadas à preservação de conteúdo digital, criada pela Library of Congress em 2010. A primeira versão do framework foi publicada em 2013 e revisada em 2019, sendo amplamente adotada por bibliotecas, arquivos e museus como ferramenta para planejamento e avaliação de capacidades de preservação digital (NDSA Levels of Preservation Working Group, 2019).

O framework organiza as atividades de preservação digital em cinco categorias funcionais: armazenamento e integridade geográfica, integridade e autenticidade dos arquivos, controle de informações, metadados e conteúdo dos arquivos.

Para cada categoria, define quatro níveis progressivos de maturidade. O nível 1 é o mais baixo, e corresponde às práticas mínimas para preservação de dados, progredindo até o nível 4, que representa práticas avançadas e sustentáveis de preservação de longo prazo. O framework não determina ferramentas ou sistemas específicos, operando somente no nível das ações e capacidades institucionais.

A relevância dos NDSA Levels para este trabalho é dupla, pois os níveis 1 e 2 do framework foram projetados para serem alcançáveis por qualquer instituição, independentemente de infraestrutura ou orçamento, estabelecendo um patamar mínimo de preservação responsável. Essa perspectiva é diretamente compatível com o contexto de instituições de memória de pequeno porte que este trabalho busca abordar. Em segundo lugar, as categorias funcionais do framework fornecem várias referências diretas para a identificação e justificativa de microsserviços essenciais apresentados na Seção 4, a partir de suas categorias de integridade dos arquivos, de metadados, de armazenamento e de conteúdo dos arquivos (NDSA Levels of Preservation Working Group, 2019).

A versão revisada de 2019 propôs práticas organizacionais, tais como políticas, delimitação de responsabilidades e documentação, como requisitos essenciais para garantir a sustentabilidade das técnicas de preservação. Para instituições de pequeno porte, essa dimensão é especialmente relevante pois mostra que a adoção de ferramentas e scripts como proposto neste trabalho deve ser acompanhada de decisões institucionais sobre periodicidade de verificações, responsabilidades de execução e documentação de procedimentos, para que o workflow produza resultados confiáveis e auditáveis ao longo do tempo.

2.4 METADADOS DE PRESERVAÇÃO DIGITAL

Dentro de uma concepção sintética, metadados são "dados sobre dados", mas essa definição mostra apenas um aspecto limitado de um conceito que deve ser considerado multidimensional. Na preservação digital e em contextos arquivísticos, os metadados são elementos estruturantes, que não apenas se limitam a descrever o conteúdo, ou descrever dados, mas englobam a maior quantidade possível de informações que terão utilidade na garantia de autenticidade, na validação de integridade e na acessibilidade dos objetos digitais preservados pelos seus usuários no futuro (Rocha, 2025b; Gilliland, 2016; Sayão, 2010)

Objetos digitais são inerentemente frágeis diante da obsolescência tecnológica, da degradação de meios e suportes, e à perda de contexto, e disso decorre a importância dos metadados de preservação digital. Documentos físicos normalmente detém informações implícitas em seus atributos físicos que carregam informações cronológicas, proveniência e autenticidade. Já os objetos digitais dependem

totalmente de informações que precisam ser agregadas ao objeto para que esse seja compreendido e preservado para uso em contextos diferentes daquele em que ele foi originalmente produzido. Isso determina que a criação e gestão desses metadados é condição essencial para qualquer estratégia de preservação digital (Caplan, 2009)

2.4.1 Categorias de metadados

Entre diversas taxonomias que a literatura propõe para classificação dos metadados, uma das mais adotadas em repositórios institucionais é a proposta por Gilliland (2016), que define um conjunto de cinco categorias funcionais para metadados: descritivos, estruturais, administrativos, técnicos e metadados de preservação. A proposta não trata essas categorias como mutuamente exclusivas, e em textos mais recentes alguns autores tratam as categorias de metadados técnicos e de preservação como subdivisões da categoria de metadados administrativos (Rocha, 2025b).

Os metadados descritivos identificam e caracterizam o conteúdo intelectual de um objeto digital, possibilitando a busca, descoberta e recuperação pelos usuários. Dentro dessa categoria incluem-se informações como título, autor, assunto, resumo, data de criação e outros identificadores persistentes. Para isso são utilizados os padrões de metadados que melhor se encaixam na natureza dos objetos tais como MARC para materiais bibliográficos, ISAD-G para acervos arquivísticos, EAD para inventários e catálogos em XML para leitura de descrições arquivísticas por máquinas; DDI e DataCite para dados de pesquisa e referência e Dublin Core "uma linguagem comum e simples para descrever qualquer tipo de recurso da Web" (Rocha, 2025b).

Os metadados estruturais documentam a organização e relações internas dos componentes de objetos digitais e são especialmente importantes em objetos compostos como livros digitalizados, coleções ou dossiês, quando a estrutura dos objetos componentes faz parte do significado do objeto composto, e é essencial para reconstituição da hierarquia e sequência lógica de seus componentes. O padrão METS (Metadata Encoding and Transmission Standard) é o esquema mais difundido para metadados estruturais.

Os metadados administrativos são informações relacionadas à administração do ciclo de vida de um objeto digital, incluindo condições de aquisição, direitos de propriedade intelectual, limitações de acesso, políticas de retenção e descarte, bem

como registros de custódia e transferência. Os metadados de direitos, que especificam permissões e restrições de uso ou reprodução do objeto são também metadados administrativos. Esta categoria de metadados normalmente é expressa por meio do padrão PREMIS Rights ou do padrão MIX para imagens digitais (Caplan, 2009).

Metadados técnicos documentam os aspectos técnicos de um objeto digital, descrevendo tipos e características técnicas dos arquivos que compõem a representação desse objeto, tais como resolução e profundidade de cor no caso de imagens, ou tipo de fonte em documentos textuais. Esses metadados são produzidos normalmente de forma automática por ferramentas como JHOVE, FITS, ExifTool e FFprobe. Também são metadados técnicos os checksums e as assinaturas digitais de arquivos. Esses últimos são especialmente importantes para verificação de integridade e autenticidade do objeto digital (Rocha, 2025b)

Por fim, os metadados de preservação são as informações de maior relevância para os objetivos de preservação digital de longo prazo. Sua principal função é documentar a história de todos os eventos custodiais do objeto digital, de forma que essas informações possam representar a autenticidade e a integridade do objeto ao longo do tempo, desde sua criação até o presente (PREMIS Editorial Committee, 2015). São os metadados que documentam a proveniência de um objeto digital e a gestão de seu arquivamento, e essas informações de proveniência são "informações sobre entidades, atividades e pessoas envolvidas na produção de um objeto digital, que podem ser usadas para formar avaliações sobre sua qualidade ou confiabilidade" (Moreau *et al*., 2013 *apud* Rocha, 2025b, p. 362)

2.5 OBJETO DIGITAL

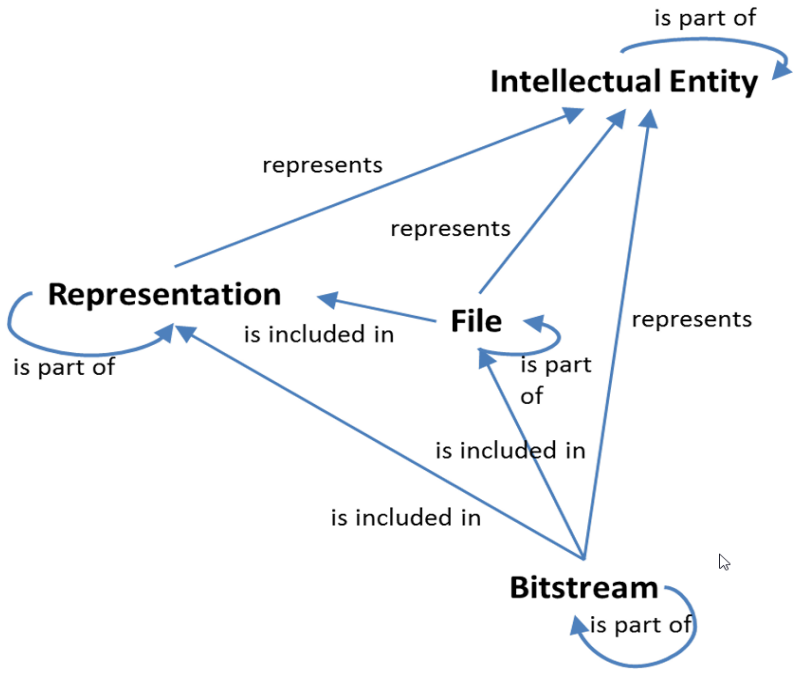
Um objeto digital pode ser descrito como uma unidade discreta de informação que está sujeita à preservação digital (PREMIS Editorial Committee, 2015). Segundo Ferreira (2006), um objeto digital é "todo e qualquer objeto de informação que possa ser representado através de uma sequência de dígitos binários (bitstream)". Isso inclui tanto os objetos nato-digitais, que são criados em um ambiente digital, quanto os objetos digitalizados, que são gerados pela conversão de formatos analógicos. Pela perspectiva da preservação digital, o objeto digital é descrito da seguinte maneira:

> [...] uma entidade complexa que envolve vários níveis de abstração (intelectual, lógico e físico) e é visto como uma entidade intelectual, que possui uma ou mais representações digitais. Essas representações digitais são formadas por arquivos digitais (Rocha, 2025b, p. 357).

Essa estrutura em vários níveis é fundamental para documentar as informações que vão permitir que o objeto permaneça acessível e recuperável através do tempo. No nível intelectual o objeto é uma criação intelectual ou artística, como um documento, um artigo ou uma fotografia, e é esse nível que oferecerá a identificação da relação entre o objeto, seus propósitos e os seus usuários. No nível lógico, o objeto é representado por arquivos digitais, codificados segundo as regras de codificação dos formatos de arquivo usados e organizados em estruturas de pastas. O nível físico representa o armazenamento efetivo dessas representações intelectuais e lógicas em um meio ou suporte físico, de rede ou em nuvem. Ainda segundo Rocha (2025b) “um objeto digital não existe sem ter suas representações digitais afixadas em um suporte”.

O modelo PREMIS também divide a descrição de objetos digitais em categorias relacionadas entre si, em definições correspondentes aos níveis de abstração de Rocha (2025b): A categoria de Entidade Intelectual representa um conjunto de conteúdo intelectual ou artístico, relevante a sua comunidade designada e considerado uma unidade intelectual para fins de gestão e descrição, que pode incluir outras entidades intelectuais. A categoria de Representação é um conjunto de arquivos que inclui os metadados estruturais que são necessários para entregar a reconstrução de uma Entidade Intelectual. A categoria de Arquivo define esse elemento como uma sequência nomeada e ordenada de bytes dentro de um padrão reconhecido por um sistema operacional, suas permissões de acesso e características como tamanho e dados cronológicos de última alteração. Por fim a categoria que define o Bitstream, que é uma sequência de dados dentro de um arquivo que podem ser contíguos ou não, possuem propriedades comuns significativas para preservação, e não podem ser transformados em um arquivo independente sem que se adicione dados estruturais ou sejam reorganizados de acordo com as definições de um formato de arquivo em particular (PREMIS Editorial Committee, 2015). A Figura 5 representa as relações conceituais entre essas quatro categorias — entidade intelectual, representação, arquivo e bitstream.

Figura 5 — Relações conceituais entre categorias de objetos digitais



Fonte: (PREMIS Editorial Committee, 2015, p. 9)

Um objeto digital pode ser definido como um componente digital, quando “é parte de um ou mais documentos digitais, incluindo os metadados necessários para ordenar, estruturar ou manifestar seu conteúdo e forma, que requer determinadas ações de preservação [...]” (CONARQ, 2020b).

2.5.1 Representações do Objeto Digital

A digitalização de um acervo gera representações digitais derivadas a partir de documentos físicos originais como objetos digitais, em arquivos. Márdero Arellano (2004) aponta que a preservação digital envolve não só o armazenamento dos arquivos, mas também a garantia de que essas representações criadas preservem os atributos essenciais do original ao longo do tempo e de mudanças tecnológicas futuras. Parâmetros técnicos como resolução, formato, método de compressão e outras configurações devem ser planejadas criteriosamente não só como decisões técnicas, mas também como decisões de preservação.

O CONARQ (2010), em suas Recomendações para Digitalização de Documentos Arquivísticos Permanentes, determina que o processo de digitalização

deve produzir pelo menos dois tipos distintos de representação para cada documento original digitalizado: uma representação de preservação que será usada para guarda de longo prazo com a máxima fidelidade possível ao documento original, e uma representação de acesso, que será destinada à consulta ou disseminação entre os usuários. Essas duas representações são criadas a partir de requisitos distintos e não raramente conflitantes, mas que foram especificados para criar o melhor resultado que se encaixe no objetivo de cada tipo de representação.

A matriz será uma representação de preservação e por isso deve ser criada com a melhor qualidade técnica possível, não podendo ser editada, comprimida com perdas ou modificada de qualquer forma após sua criação. O formato TIFF é altamente recomendado para imagens estáticas, pois é um formato aberto, sem compressão destrutiva, e é amplamente compatível com ferramentas de preservação. É na matriz que serão realizadas as ações de monitoramento e de preservação digital do repositório, e é a partir da matriz preservada que serão criadas novas representações, como derivadas, caso as representações derivadas atuais se tornem obsoletas ou incompatíveis com tecnologias de acesso no futuro (CONARQ, 2010; Rocha, 2025b).

As representações derivadas serão produzidas a partir de uma matriz preservada, em um processo de conversão onde serão aplicados parâmetros técnicos específicos à finalidade de uso ou acesso, de acordo com as necessidades de consumo do usuário. Uma derivada de acesso, poderia, por exemplo, ser gerada como um arquivo PDF com reconhecimento ótico de caracteres (OCR), um formato que permite a pesquisa de conteúdo e fácil acesso eletrônico. Uma derivada de navegação também seria gerada a partir da matriz, porém com outros parâmetros técnicos, para produzir, por exemplo, uma imagem de baixa resolução utilizada como miniatura em interfaces de pesquisa. Essa estratégia de separação entre representações de preservação e representações derivadas oferece a importante capacidade de gerar novas derivadas a partir da matriz sempre que for necessário, impedindo que a degradação de cópias ou a obsolescência de formato afete a integridade do acervo (CONARQ, 2010).

No modelo OAIS essas diferentes representações são relacionadas diretamente a tipos de pacotes de informação. A matriz faz parte do Pacote de Informação de Arquivamento (AIP), mantido sob custódia permanente pelo repositório, e as representações derivadas, geradas a partir dela, podem integrar o Pacote de Informação de Arquivamento (AIP) como representações adicionais, podendo ainda

compor, sob demanda, um Pacote de Informação de Disseminação (DIP) para uso e acesso (CCSDS, 2012).

2.5.2 Propriedades Significativas

É necessário determinar quais atributos ou características dos objetos digitais originais que não podem ser perdidos durante os processos de preservação, para não degradar o valor das representações digitais nesses processos, como em uma migração de formato, por exemplo. Esses atributos são chamados de propriedades significativas, e a definição desses atributos varia conforme a natureza do objeto e as expectativas de sua comunidade de usuários (Dappert; Farquhar, 2009).

A fonte ou tipo de letra utilizado pode não ser uma propriedade significativa em um documento administrativo como uma ata de reunião, pois nesse caso o conteúdo textual é o que importa preservar. Já em uma publicação histórica como um jornal antigo, a mesma propriedade pode ser indispensável na preservação do conteúdo, pois os aspectos tipográficos fazem parte da identidade editorial e são parte essencial do valor documental. Isso determina que processos como uma migração de formato devem ser considerados de forma diferente para cada categoria de objeto, respeitando o que foi estabelecido no plano de preservação do repositório ou da instituição (Dappert; Farquhar, 2009).

2.5.3 Empacotamento do Objeto Digital

Um objeto digital não é custodiado de forma isolada, ele é sempre armazenado e gerenciado através de uma estrutura lógica denominada Pacote de Informação, que reúne todas as representações e todos os metadados que são necessários para que esse objeto possa ser identificado, validado, transformado e acessado ao longo do tempo. O Pacote de Informação é definido assim:

> Um recipiente lógico (container) para uma entidade intelectual, suas representações, arquivos e cadeias de bits. Inclui metadados descritivos, estruturais e administrativos. Envolve informação de representação, informações de preservação, informações de empacotamento e informação descritiva (Rocha, 2025a).

O modelo OAIS identifica três tipos de Pacote de Informação, sendo cada um deles relacionado com uma das etapas do ciclo de vida do objeto no repositório. O Pacote de Informação de Submissão (PIS) é produzido pela entidade que fornece o conteúdo ao repositório, e inclui o objeto digital na forma em que foi gerado ou digitalizado e todos os metadados disponíveis ou produzidos naquele momento. Após o processo de ingestão, verificação e validação pelo repositório, o conteúdo é reestruturado e complementado com metadados adicionais. Esse novo conjunto resulta na criação do Pacote de Informação de Arquivamento (PIA), que representa a forma canônica de armazenamento permanente do objeto. O Pacote de Informação de Disseminação (PID) é criado sob demanda a partir do PIA, utilizando representações e metadados que melhor atendam às necessidades particulares dos usuários (CCSDS, 2012).

Para viabilizar a interoperabilidade entre repositórios e ferramentas de preservação é necessário que a estrutura interna dos pacotes de objetos digitais seja padronizada. Dentro desse contexto o projeto europeu E-ARK criou o Common Specification for Information Packages (CSIP), uma especificação que define uma arquitetura comum para os três tipos de pacotes do modelo OAIS, baseada no padrão de metadados estruturais METS e compatível com o padrão PREMIS de metadados de preservação. O objetivo do CSIP é oferecer um padrão para que pacotes criados em uma instituição possam ser recebidos, processados e interpretados por outros sistemas de outras instituições sem a necessidade de conversões ou transformações, uma condição particularmente importante em situações de transferência de custódia de acervos ou colaboração entre instituições (Andrade, 2024; DILCIS Board, 2021).

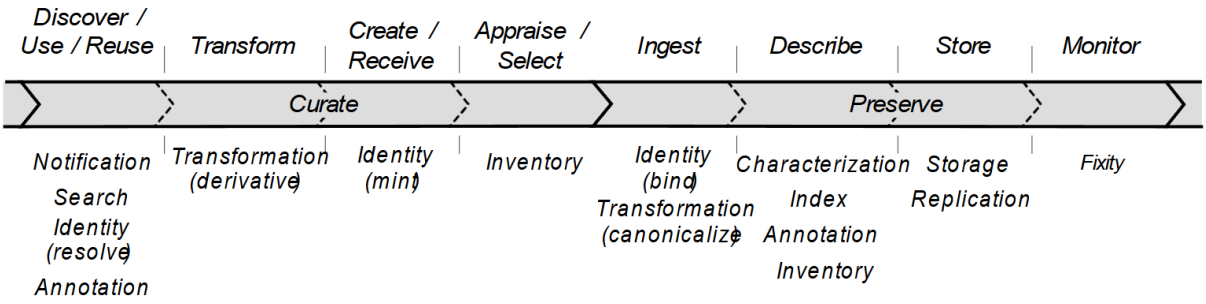
A definição criteriosa da estrutura e conteúdo do pacote de informação, suas representações, metadados, estruturas de pastas e arquivos, e regras de nomenclatura, deve acontecer antes do início das atividades de produção e digitalização, e não depois. Rocha (2025b, p. 369) aponta que a etapa de descrição e representação no processo de digitalização resulta também na especificação dos pacotes de submissão, armazenamento e disseminação, que são os recipientes dos objetos digitais. Isso é relacionado diretamente com a arquitetura de preservação definida pelo modelo OAIS, que indica que essas especificações devem ser realizadas idealmente antes da digitalização, na etapa de planejamento de preservação. Um dos fatores que podem comprometer a proveniência e a autenticidade dos objetos digitais

é tratar o empacotamento como uma etapa secundária e posterior, pois assim as decisões sobre estrutura e metadados não são documentadas desde o início.

2.6 MICROSSERVIÇOS DE PRESERVAÇÃO DIGITAL

O conceito de microsserviço tem origem na arquitetura de software, em que são definidos pequenos componentes funcionais independentes, especializados e projetados para executar tarefas específicas. Essa abordagem foi adaptada ao domínio da preservação digital pelo University of California Curation Center (UC3), que na primeira década dos anos 2000 passou a estruturar seus processos de curadoria com base nesse paradigma. A proposta essencial dessa adaptação, desenvolvida por Abrams, Kunze e Loy (2010), é de que é possível separar as cadeias de ações necessárias à preservação de objetos digitais em pequenas unidades funcionais, sendo cada ação responsável por uma tarefa específica, sendo possível combinar essas unidades de diversas maneiras para criar fluxos flexíveis e adaptáveis aos projetos de preservação digital de uma instituição. A Figura 6 ilustra a aplicabilidade de microsserviços ao longo do ciclo de vida de curadoria e preservação.

Figura 6 — Aplicabilidade de microsserviços no ciclo de vida de curadoria e preservação



Fonte: Abrams; Kunze; Loy, 2010

É também na arquitetura de software que encontramos características dos microsserviços, onde Lewis e Fowler (2014) definiram que um microsserviço deve ser substituível e atualizável, e que pode ser implementado e mantido sem afetar qualquer outro elemento ou parte do sistema ou fluxo do qual faça parte.

No contexto da preservação digital, um microsserviço é uma unidade funcional independente e reutilizável, que executa uma ação específica sobre um objeto digital

ou metadados que estão a ele associados. Em sistemas integrados monolíticos de preservação digital a ação desse microsserviço normalmente estará integrada dentro de um fluxo pré-definido, normalmente inflexível. Em uma arquitetura baseada em microsserviços, cada ação é um componente independente que pode ser desenvolvido, mantido e processado separadamente. Essa independência oferece uma flexibilidade operacional que sistemas integrados não oferecem, beneficiando especialmente aqueles contextos institucionais onde os recursos tecnológicos e humanos são limitados (Abrams; Kunze; Loy, 2010).

Van Garderen (2010) afirma que, pelo tamanho reduzido e pela natureza autocontida, os processos de desenvolvimento, implementação e manutenção de microsserviços são mais simples, além de serem mais facilmente substituídos quando necessário. Essa particularidade é sintetizada na máxima “preservar objetos, não sistemas” (Van Garderen, 2010), que ecoa o princípio “objetos permanentes, sistemas descartáveis” formulado pelo UC3 (California Digital Library, 2010). Dentro desse conceito, os objetos digitais e os pacotes de informação são entidades cuja existência é totalmente independente das ferramentas de software que os manipulam, o que produz grande resiliência em relação à obsolescência tecnológica. Adicionalmente, Abrams, Kunze e Loy (2010) afirmam que, apesar de cada microsserviço ter um escopo bastante definido que caracteriza sua atomicidade, é possível que funções complexas de curadoria e preservação surjam da combinação estratégica de serviços individuais encadeados em pipelines de processamento.

3 METODOLOGIA

Esta pesquisa tem natureza qualitativa, buscando compreender e relacionar conceitos e modelos da área de preservação digital. Quanto aos objetivos, a pesquisa caracteriza-se como exploratória e descritiva. É exploratória pois investiga uma área ainda pouco explorada no contexto de instituições de memória de pequeno porte; e descritiva porque especifica características e capacidades de aplicativos de software e seu funcionamento dentro de um workflow proposto. Trata-se também de uma pesquisa aplicada de caráter tecnológico, pois busca como resultado final um conjunto de conhecimentos práticos dentro de um workflow com aplicativos selecionados e scripts de código, buscando a resolução de um problema específico que ocorre em instituições de preservação de pequeno porte, ou seja, a carência de soluções acessíveis e simples para implementar microsserviços essenciais de preservação digital.

3.1 CARACTERIZAÇÃO DO CENÁRIO INSTITUCIONAL: A INSTITUIÇÃO DE MEMÓRIA DE PEQUENO PORTE

A caracterização do pequeno porte em instituições de memória ocupa lugar central neste trabalho, pois é o cenário institucional para o qual o workflow proposto se destina. Esta seção apresenta definições de aspectos operacionais dessa categoria e estabelece os parâmetros de porte que justificam a criação do workflow proposto. Cabe registrar que a literatura especializada em preservação digital não oferece uma definição normativa de "instituição de memória de pequeno porte". A maioria dos modelos conceituais de referência foi elaborada com base na realidade de grandes instituições, especialmente nos contextos europeu e norte-americano (CCSDS, 2012; Higgins, 2008; California Digital Library, 2010), sem abordar especificidades relativas a porte. Diante desse vazio bibliográfico, a caracterização apresentada nesta seção foi construída pelo autor a partir da observação direta de instituições de diversos portes durante as atividades de visitas institucionais e estágios realizados durante o curso de graduação, identificando um conjunto recorrente de aspectos que distinguem as instituições de menor capacidade institucional das demais. Tais aspectos serão adotados como critérios operacionais de caracterização neste trabalho.

O conjunto das instituições brasileiras dedicadas à custódia e conservação de acervos documentais que compreende arquivos, bibliotecas e museus apresenta significativa heterogeneidade. Em um extremo encontramos instituições governamentais com equipes especializadas, infraestrutura computacional dedicada e políticas formais de preservação, como o Arquivo Público do Rio Grande do Sul, o Arquivo Histórico do Rio Grande do Sul e o Arquivo Judicial Centralizado do TJRS. Ao mesmo tempo, em outro extremo encontramos arquivos comunitários, centros documentais universitários e acervos de associações civis que custodiam parte importante do patrimônio documental brasileiro em condições estruturais completamente diversas. O Relatório Técnico sobre Censo de Arquivos Públicos Instituídos no Brasil (CONARQ, 2020a) evidencia esse desequilíbrio ao demonstrar que parte expressiva dos municípios brasileiros não dispõe de arquivo público estruturado. A aplicação integral dos modelos conceituais de preservação digital a esse contexto ignora não apenas diferenças materiais como limitações orçamentárias e de pessoal, mas também diferenças qualitativas que envolvem graus distintos de maturidade organizacional, formalização de processos e capacidade tecnológica. Cinco aspectos, observados de forma recorrente nas instituições de menor capacidade, são adotados como critérios operacionais de caracterização e analisados a seguir:

O primeiro aspecto é o de recursos humanos. As instituições de pequeno porte costumam funcionar com equipes limitadas, muitas vezes constituídas por apenas um profissional encarregado do acervo, ocasionalmente com alguns auxiliares, sem delimitação precisa entre diferentes funções técnicas. É frequente que um mesmo profissional acumule responsabilidades de catalogação, preparação, administração e digitalização. Grácio (2012, p. 74) destaca que a implementação da preservação digital exige especialistas de várias áreas e que a ausência de equipe multidisciplinar capacitada constitui um dos principais obstáculos organizacionais para a viabilidade do processo.

O segundo aspecto é o financeiro-orçamentário. Essas instituições caracterizam-se por orçamentos reduzidos ou muito limitados e por baixa previsibilidade de recursos. Os recursos para projetos de digitalização e preservação digital frequentemente dependem de editais externos, repasses esporádicos ou parcerias temporárias, o que dificulta o planejamento de ações de preservação sustentadas ao longo do tempo.

O terceiro aspecto é o da infraestrutura tecnológica. O cenário típico envolve estações de trabalho de uso geral, armazenamento local em discos rígidos comuns, ausência ou precariedade de soluções de backup sistemático e ausência de servidores dedicados exclusivamente à preservação digital. Esse cenário contrasta com os requisitos exigidos por ferramentas como Archivematica, Preservica e Libnova Libsafe, cuja abrangência funcional se traduz em arquiteturas que demandam servidores dedicados, configuração avançada e manutenção contínua por profissionais especializados, ou contratos de Software as a Service de custo elevado.

Considera-se de pequeno porte, sob este aspecto, a instituição que não dispõe de servidores dedicados à preservação digital, não conta com profissionais especializados na área de tecnologia da informação em seu quadro fixo e opera seus processos digitais em equipamentos de uso geral.

O quarto aspecto diz respeito à relação entre o porte do acervo e a capacidade institucional. As instituições de pequeno porte não necessariamente custodiam acervos de volume reduzido: A observação de arquivos de instituições civis locais revelou casos em que quantidade expressiva de documentação histórica relevante era mantida em condições estruturais precárias. A caracterização do pequeno porte não se dá, portanto, pelo volume absoluto do acervo, mas pela desproporção entre o volume custodiado e a capacidade institucional de gerenciá-lo e preservá-lo. É precisamente essa desproporção que o workflow proposto busca reduzir.

O quinto aspecto é o da maturidade organizacional. Instituições de menor porte geralmente não possuem políticas formais de preservação digital, planos de preservação documentados, padrões institucionais para descrição arquivística ou rotinas sistemáticas de monitoramento do acervo. A gestão da preservação funciona de maneira informal, fundamentada no conhecimento individual de profissionais cuja dedicação não é exclusiva à preservação.

Esses cinco aspectos configuram o que este trabalho define operacionalmente como instituição de memória de pequeno porte. O conceito não implica ausência completa de capacidade, mas operação em regime de capacidade reduzida e sem a implementação integral dos modelos conceituais de referência. Cabe precisar o sentido de "porte" aqui adotado: o termo não designa o tamanho da instituição em sentido genérico — número de funcionários, volume de acervo ou abrangência administrativa — mas a conjugação dos cinco aspectos descritos. Entre eles, a infraestrutura de TI é o aspecto sobre o qual o workflow proposto incide de modo mais

direto, ao oferecer uma solução exequível em equipamentos de uso comum e com software livre; os demais aspectos delimitam o contexto de capacidade reduzida que torna essa parcimônia necessária. Decorre daí que a categoria não se restringe a instituições pequenas em sentido absoluto: nela também se enquadram instituições de maior porte cujo aparato de TI dedicado à preservação digital seja igualmente limitado, pois o que define o cenário de aplicação é a desproporção entre a capacidade institucional — sobretudo computacional — e as demandas de preservação, e não o porte institucional em sentido amplo.

É importante também demarcar o limite inferior dessa caracterização: recursos reduzidos não equivale à ausência de recursos. A exequibilidade do workflow pressupõe um patamar mínimo de infraestrutura de pessoal e de recursos computacionais — uma pessoa com razoável conhecimento do sistema operacional Linux, e um equipamento de uso geral, porém capaz de executar o ambiente Linux e os aplicativos de software livre que operacionalizam os microsserviços — de modo que instituições desprovidas até desse mínimo ficam fora do cenário de aplicação da proposta, pois nelas o workflow não teria possibilidade de execução bem-sucedida. A caracterização delimita, assim, uma faixa de capacidade reduzida, porém suficiente, e não a sua inexistência. Valores de referência para esta capacidade mínima podem ser encontrados na seção 8, na configuração da máquina virtual usada na avaliação do workflow.

É esse perfil que delimita o cenário de aplicação da pesquisa e orienta as decisões metodológicas das seções seguintes: a condição de governança do critério de essencialidade dos microsserviços (Seção 4), que pressupõe a inexistência de políticas formais de preservação, e os critérios de seleção das ferramentas de software livre (Seção 5), voltados à exequibilidade em equipamentos de uso comum.

3.2 PERCURSO METODOLÓGICO

O percurso metodológico está organizado em torno dos objetivos específicos da pesquisa, conforme sintetizado no Quadro 1 a seguir:

Quadro 1 — Percurso metodológico por objetivo

Nº	Objetivo	Percurso Metodológico	Fontes
1	Identificar os microsserviços essenciais de preservação digital para instituições de memória de pequeno porte.	Pesquisa bibliográfica e documental que busca identificar os microsserviços de preservação digital.	Publicações e manuais de software técnicos sobre microsserviços de preservação.
2	Analisar aplicativos e bibliotecas de software livre, disponíveis na plataforma Linux, que apoiam ou realizam a implementação de microsserviços essenciais de preservação digital.	Identificar aplicativos que realizam funções que implementam microsserviços de preservação digital; Caracterizar os aplicativos em relação ao seu atendimento aos microsserviços.	Publicações e Manuais de Software técnicos sobre microsserviços de preservação digital; Aplicativos que realizam microsserviços de preservação digital e suas documentações
3	Propor uma arquitetura para implementação de um workflow para a execução dos microsserviços essenciais de preservação digital na plataforma Linux.	Identificar fluxos; Propor workflow.	Resultados das análises anteriores.
4	Implementar, por meio de scripts Python, a orquestração do workflow e o encapsulamento dos microsserviços, realizados por aplicativos e bibliotecas disponíveis no ambiente Linux.	Projetar scripts; Escrever o código dos scripts; Testar scripts; Integrar scripts no workflow.	Resultados das análises anteriores.
5	Avaliar a execução do workflow e seus resultados, sobre elementos de um acervo de objetos digitais obtidos no corpus do Projeto Format-Corpus da Open Preservation Foundation (OPF).	Executar o workflow e seus scripts em elementos de acervo.	Resultados das análises anteriores; Acervo.

Fonte: elaborado pelo autor.

Em relação ao objetivo 1, a identificação dos microsserviços essenciais de preservação digital envolveu pesquisa bibliográfica e documental sobre oito modelos conceituais de curadoria e preservação digital — OAIS, DCC, PLANETS, Dappert-Farquhar, UC3, Pavão-Caregnato-Rocha, InterPARES e NDSA Levels of Digital Preservation. Desses modelos foram extraídas as atividades recorrentes passíveis de caracterização como microsserviços, na acepção de Abrams, Kunze e Loy (2010), e apurou-se a recorrência de cada candidato, adotando-se a estrutura de categorias funcionais de Pavão, Caregnato e Rocha (2016).

Cada atividade foi então classificada através de três condições: recorrência (presença em três ou mais dos oito modelos); escopo (pertencimento às categorias de preservação ingestão, preservação, transformação e armazenamento, excluídas as funções de seleção, acesso, uso e monitoramento); e governança (capacidade de

operar no patamar mínimo de preservação responsável, sem pressupor ou depender da existência de política formal de preservação).

Classificaram-se como essenciais os microsserviços que satisfizeram as três condições; os que satisfizeram a recorrência mas falharam em escopo ou em governança foram classificados como contextuais — igualmente essenciais ao ciclo de curadoria, porém com execução condicionada a um contexto institucional específico, que não se encaixa na categorização da instituição de pequeno porte — registrados para orientar a evolução futura e não considerados para integrar a arquitetura do workflow mínimo.

Tendo como base os microsserviços essenciais identificados no objetivo 1, o objetivo 2 envolveu identificar aplicativos e bibliotecas de software livre da plataforma Linux que apoiam a implementação desses microsserviços. Para cada microsserviço, os candidatos levantados na literatura técnica e na documentação dos aplicativos foram confrontados com seis critérios, derivados dos cinco aspectos que caracterizam a instituição de memória de pequeno porte (Seção 3.1):

(C1) licenciamento livre ou de código aberto;

(C2) baixo consumo de recursos e execução em hardware comum;

(C3) instalação simples por gerenciador de pacotes e dependências mínimas;

(C4) operação automatizável por linha de comando e integrável a scripts Python;

(C5) adequação funcional ao microsserviço;

(C6) maturidade e manutenção ativa.

Adotou-se, para cada microsserviço, a ferramenta que melhor satisfaz o conjunto, registrando-se as demais como alternativas documentadas; em caso de equivalência, prevaleceu a de menor número de dependências. A análise e a síntese das ferramentas estão desenvolvidas na seção 5.

O objetivo 3 envolveu propor uma arquitetura para esses microsserviços dentro de um workflow, organizada segundo estes três princípios: sequencialidade com dependências explícitas, atomicidade dos microsserviços e parcimônia. Os microsserviços de preservação digital foram caracterizados como etapas de um fluxo de trabalho, indicando-se aplicativos, ferramentas e bibliotecas disponíveis no ambiente Linux que operacionalizassem cada uma dessas etapas. A especificação do workflow teve como principais requisitos permitir a execução de fluxos independentes, abrangendo diferentes fases do ciclo de vida da curadoria do objeto digital — como

ingestão e manutenção — além de assegurar o tratamento de falhas que possam ocorrer na execução dos microsserviços, dentro de um processo de orquestração.

O workflow estruturou-se em dois fluxos independentes: Um fluxo de ingestão, com nove etapas (E1 a E9) distribuídas em três fases (recepção e validação; caracterização e empacotamento; armazenamento), e um fluxo de manutenção, com uma etapa periódica (E10). Como parte fundamental da preservação digital para garantia de autenticidade, custódia e verificação de integridade, o registro de eventos de preservação (PREMIS) foi tratado como função transversal, executada ao final de cada etapa que realiza uma ação de preservação.

Foram definidos três pontos de decisão em falhas (E2, E3 e E9), que controlam a interrupção ou o prosseguimento do fluxo. Cada etapa foi caracterizada por:

a) código e nome;
b) o microsserviço que ela usa (conforme a Seção 5);
c) as ferramenta(s) Linux usadas em sua operacionalização;
d) o gatilho ou entrada esperada;
e) e a saída produzida, incluindo as informações de preservação geradas (metadados técnicos, eventos PREMIS e logs).

A análise e a síntese das etapas do workflow estão desenvolvidas na seção 6.

O objetivo 4 envolveu implementar, em linguagem Python, os scripts que realizam os microsserviços no workflow, especificados no objetivo 3. A implementação do objetivo 4 não consistiu em programar cada função de preservação a partir do zero, mas em integrar ferramentas de software livre já consolidadas, heterogêneas entre si quanto ao modo de invocação, às dependências e aos formatos de saída de seus resultados. A estratégia adotada foi implementar cada microsserviço como um script que encapsula a chamada à ferramenta ou à biblioteca Linux correspondente pelo módulo subprocess ou por interfaces nativas em Python, quando disponíveis. Assim foi possível padronizar e uniformizar a interface de invocação e a convenção de retorno dos microsserviços, de modo que, para o restante do sistema, todos se comportem da mesma maneira, recebendo as entradas esperadas e devolvendo um código de retorno padronizado, independentemente de como funciona a ferramenta ou aplicativo subjacente. É essa uniformidade que torna os microsserviços intercambiáveis e reutilizáveis em várias etapas, concretizando o princípio de atomicidade adotado para a arquitetura do workflow.

Para controlar esses scripts independentes, a implementação incluiu o desenvolvimento de um mecanismo orquestrador, também feito através de um script Python, com a função de gerenciar as etapas de execução do workflow, com o requisito de receber uma solicitação de execução a partir de parâmetros de arquivos de entrada, destinos de saída e replicação, e a seguir determinar quais etapas e microsserviços devem ser executados, encadear os scripts dos microsserviços na sequência das etapas com dependências explícitas, de modo que nenhuma etapa se inicie antes da conclusão da anterior e tratar pontos de decisão e situações de exceção, conforme o fluxo especificado. O orquestrador também teve como requisito a implementação de registrar cada atividade do fluxo em registro de logs. A análise e a síntese dos scripts estão desenvolvidas na seção 7.

O objetivo 5 envolveu avaliar o workflow sobre elementos de um acervo. Em coerência com a natureza qualitativa, aplicada e de caráter tecnológico da pesquisa, adotou-se uma avaliação funcional voltada a verificar a executabilidade do artefato e a validade dos produtos que ele gera. A avaliação organizou-se em torno de três aspectos: executabilidade, integração e adequação aos requisitos de preservação (integridade, autenticidade, rastreabilidade e redundância), que foram desdobrados em nove critérios objetivos (CR1 a CR9), tomando como patamar de referência os níveis 1 e 2 dos NDSA Levels of Digital Preservation.

O workflow foi executado em uma máquina virtual que reproduziu uma configuração de hardware de uso comum, sem softwares que exigem licenciamento (utilizando VirtualBox, Ubuntu LTS, ferramentas e bibliotecas), cuja configuração e versões foram registradas no quadro 19.

A avaliação foi conduzida sobre cinco lotes de teste, compostos por objetos do Format-Corpus da Open Preservation Foundation e por itens de controle:

a) um lote íntegro que testou o fluxo nominal (E1 a E9);
b) um lote misto que testou alertas não-fatais como o não reconhecimento de formato, a não conformidade de formato e falhas pontuais de derivação, todos sem interrupção;
c) um lote com simulação de vírus que testou o primeiro ponto de decisão (E2);
d) um lote corrompido, que testou o segundo ponto de decisão (E3), por meio de um SIP BagIt com conteúdo adulterado; e

e) um lote íntegro que testou o terceiro ponto de decisão (E9) através de um destino de replicação inacessível.

A etapa E10 (verificação periódica de integridade) foi acionada no seu contexto normal, de forma isolada, sobre pacotes AIP já armazenados. Na execução de cada lote de teste, foram coletadas evidências a partir da saída de terminal, dos logs, da estrutura dos pacotes AIP gerados, dos documentos PREMIS gerados, dos relatórios de identificação e conformidade e a validação BagIt, e foi feita a avaliação a partir da consolidação de atendimento aos critérios no quadro 46.

4 MICROSSERVIÇOS DE PRESERVAÇÃO DIGITAL PARA INSTITUIÇÕES DE MEMÓRIA DE PEQUENO PORTE

A identificação dos microsserviços essenciais de preservação digital para instituições de memória de pequeno porte foi realizada através de pesquisa bibliográfica e documental, com base nos modelos conceituais que foram apresentados no referencial teórico. A pesquisa buscou extrair e relacionar ações ou atividades recorrentes entre esses modelos que poderiam ser caracterizadas e identificadas como microsserviços dentro da definição de microsserviço proposta por Abrams, Kunze e Loy (2010). Segundo os autores, microsserviços são unidades funcionais independentes, com escopo claramente definido, e capazes de executar ações específicas em objetos digitais ou sobre metadados associados a esses objetos digitais.

Conforme descrito na metodologia, a identificação dos microsserviços essenciais foi realizada a partir análise de atividades presentes nos modelos ou frameworks OAIS, DCC, PLANETS, Dappert-Farquhar, UC3, Pavão-Caregnato-Rocha, InterPARES e NDSA Levels of Digital Preservation, que possivelmente realizam microsserviços de preservação digital.

Neste trabalho, o critério usado para identificar a essencialidade de microsserviços combina três condições: recorrência, escopo e governança. Essas condições devem ser satisfeitas simultaneamente para que um microsserviço seja classificado como essencial e integre o conjunto de microsserviços que serão incluídos no workflow.

A primeira condição é a recorrência entre os modelos conceituais. Cada atividade foi analisada quanto à sua presença, de forma central ou explícita, nos oito modelos e frameworks. Adotou-se como limiar de recorrência a presença da atividade em três ou mais desses modelos, uma medida de frequência que indica convergência suficiente da literatura para caracterizá-la como função importante da preservação digital, e não como particularidade idiossincrática de um único ou poucos modelos. A apuração da recorrência é apresentada no quadro 2.

A segunda condição é o escopo. As atividades foram analisadas para determinar se a função se encaixa nas categorias conceituais que foram consideradas como categorias de preservação digital que lidam especificamente com as matrizes originais de objetos digitais, no recorte deste trabalho: ingestão, preservação,

transformação e armazenamento. Atividades cuja função se encaixe em categorias fora desse recorte, como acesso, uso, monitoramento de obsolescência ou outras que não são atividades específicas de preservação de matrizes, não satisfazem essa condição e microsserviços para essas funções não são considerados para o workflow.

A terceira condição é a governança. Para essa condição as atividades tiveram sua função analisada para determinar sua capacidade de operar no patamar mínimo de preservação responsável, pressupondo a inexistência de políticas formais de preservação, como é típico em muitas instituições de pequeno porte. Se a função exigir qualquer tipo de política formal preexistente, ou interrupção de qualquer tipo para tomada de decisões de preservação, ela não satisfaz esta condição.

A combinação dessas três condições produz as classificações registradas nas colunas de recorrência, escopo e governança do quadro 3. Um microsserviço é classificado como essencial quando satisfaz o limiar de recorrência e as duas condições, de escopo e governança, constituindo uma função essencial sem a qual a preservação digital não se sustenta nem mesmo em seu patamar mínimo dentro do escopo deste trabalho, a instituição de pequeno porte.

O microsserviço é classificado como contextual quando satisfaz a recorrência, mas enfrenta restrição em pelo menos uma das condições de escopo ou governança, restrição que reflete a dependência de um contexto institucional que não se encaixa na caracterização da instituição de pequeno porte. São microsserviços igualmente essenciais ao ciclo de curadoria, cuja execução, contudo, fica condicionada a dependências de escopo ou governança que não são normalmente encontradas no contexto da instituição de pequeno porte, razão pela qual não integram o núcleo do workflow proposto nas seções seguintes, sendo registrados como referências para possível evolução futura, conforme a maturidade institucional progrida.

4.1 MICROSSERVIÇOS DE PRESERVAÇÃO DIGITAL NOS MODELOS CONCEITUAIS

Esta seção analisa os microsserviços de preservação digital presentes nos modelos conceituais OAIS, DCC, PLANETS, Dappert-Farquhar, UC3, Pavão-Caregnato-Rocha, InterPARES e NDSA Levels of Digital Preservation.

4.1.1 Microsserviços no modelo OAIS

O modelo OAIS (apresentado na seção 2.3.1), por ser um modelo de referência amplamente usado para repositórios digitais e por especificar as principais funções (entidades funcionais) a serem realizadas por um repositório, é referência essencial e ponto de partida para este estudo de identificação de microsserviços de preservação digital. As entidades funcionais que ele define — Ingestão, Armazenamento, Gerenciamento de dados, Administração, Planejamento de preservação e Acesso — são conjuntos de atividades que podem ser analisadas e separadas em ações menores independentes, que por sua vez podem ser definidas em microsserviços.

Na entidade funcional de Ingestão, observam-se microsserviços relacionados ao recebimento, verificação e validação dos objetos digitais fornecidos pelo produtor, além da criação do pacote de Informação de Arquivamento (AIP) a partir do Pacote de Submissão (SIP). A entidade de ingestão não corresponde a um único microsserviço, mas a uma categoria funcional que reúne ações distintas como a verificação de admissibilidade do material recebido (pelo resultado da varredura antivírus, entre outras) e a constituição ou validação do pacote de informação de entrada, ações que o próprio modelo descreve em sua função de quality assurance sobre o conteúdo submetido. A entidade de armazenamento está relacionada e traz fundamentos para microsserviços de armazenamento em mídias e de verificação de integridade. Um microsserviço de extração de metadados teria sua fundamentação na entidade de Gerenciamento de Dados.

A entidade de Planejamento de Preservação traz fundamentos para microsserviços de monitoramento de formatos e de avaliação de risco de obsolescência. Um microsserviço de geração de representações derivadas seria fundamentado na ação de Transformação do OAIS, onde são produzidas novas representações a partir do conteúdo original como ação de preservação. A entidade de Acesso, distintamente, fundamenta um microsserviço de geração do pacote de disseminação (DIP), para produção de derivadas de acesso ou uso. A arquitetura dos pacotes de Informação como definida no modelo OAIS (SIP, AIP, DIP) também seria determinante para definição de um microsserviço de empacotamento, que organizaria objetos digitais e seus metadados em um recipiente padronizado, condição essencial para gestão de longo prazo do acervo.

A arquitetura informacional do OAIS prevê ainda, no conjunto de Informação de Descrição da Preservação (PDI), a Informação de Proveniência, que documenta a história da custódia e as ações executadas sobre o conteúdo ao longo do tempo, o que fundamentaria diretamente um microsserviço de registro de eventos de preservação.

4.1.2 Microsserviços no modelo DCC de ciclo de vida da curadoria digital

O Modelo DCC, conforme apresentado na seção 2.3.2, apresenta ações de curadoria digital ao longo de um ciclo de vida, organizadas em três categorias: ações sequenciais, ações que são executadas ao longo de todo o ciclo de vida e ações executadas ocasionalmente.

Dentro dessas três categorias, além das ações do ciclo de vida completo (descrição, representação e planejamento da informação, acompanhamento das atividades da comunidade, curar e preservar), e além das ações ocasionais (descartar, reavaliar e migrar), o ciclo de vida de um objeto digital é descrito em estágios iterativos e sequenciais (conceitualizar, criar e receber, avaliar e selecionar, ingerir, ação de preservação, armazenar, acessar, usar, reusar e transformar). Estas ações sequenciais contribuem para identificar microsserviços que precisam ser executados em uma sequência determinada e lógica, e serem repetidos ao longo do tempo.

Nas ações sequenciais, após a conceitualização, as ações de Criar ou Receber, Avaliar e Selecionar e Ingerir correspondem às ações iniciais do ciclo. Essas ações relacionam-se diretamente a microsserviços de validação e empacotamento. O DCC também contempla a ação de Observação da comunidade designada, uma ação transversal de acompanhamento da comunidade de usuários (Seção 2.3.2).

As ações de Preservar e Armazenar fundamentam um microsserviço de armazenamento com replicação, enquanto a ação de transformar — que no ciclo do DCC consiste em criar novos dados a partir de dados originais para, por exemplo, migração de formato — fundamentam um microsserviço de geração de representações derivadas. As ações de Acessar, Usar e Reutilizar apontam para um microsserviço de geração de derivadas DIP.

O modelo também reforça a importância de ações de descrição e representação dos objetos digitais, que ocorrem em ações tanto na categoria das

Ações de ciclo de vida completo, quanto nas ações sequenciais, nas ações de Planejamento da Preservação e Ação de Preservação, respectivamente, apontando para microsserviços de extração de metadados e de registro de eventos ao longo do fluxo de preservação e de validação de conformidade de formato. Higgins (2008) descreve que as ações de preservação compreendem, entre outras, a garantia de estruturas de dados e formatos de arquivo aceitáveis. Isso sustenta o microsserviço de verificação de conformidade de formato como microsserviço identificado.

4.1.3 Microsserviços no projeto PLANETS

Conforme apresentado na seção 2.3.3, o modelo funcional de PLANETS é estruturado em funções de observação, planejamento da preservação, ação de preservação e de caracterização. A abordagem do projeto PLANETS distingue atividades como a characterisation/caracterização, que corresponde a processos de identificação de formatos e de extração de propriedades e características técnicas de objetos digitais, e também distingue atividades de Preservation Action/Ação de preservação, que incluem conversões, migrações de formato e emulação (Farquhar; Hockx-Yu, 2007).

É importante notar que a própria characterisation, no modelo PLANETS, não é uma operação única. Ela compreende ao menos dois processos conceitualmente distintos: a identificação de formato, que determina qual é o formato de cada arquivo, e a validação de conformidade, que verifica se o arquivo está bem-formado e em conformidade com a especificação do formato declarado. Essa distinção é relevante para a identificação dos microsserviços, pois identificar um arquivo como sendo um PDF é uma operação diferente de atestar que esse PDF é um PDF/A válido segundo a norma.

O PLANETS também estrutura a atividade de planejamento de preservação através de sua ferramenta Plato, que analisa objetos digitais para avaliar estratégias diante de obsolescência tecnológica, o que fundamentaria, ainda que indiretamente, o microsserviço de monitoramento de formatos. As ações de preservação do projeto através da migração de formatos fundamentaria um microsserviço de geração de representações derivadas, entendida como a produção de novas representações a partir da matriz preservada.

No contexto das instituições de pequeno porte, a identificação de formato, a extração de metadados técnicos e a validação de conformidade de formato, são funções derivadas da *characterisation* e consideradas essenciais. É necessário, porém, delimitar com precisão o escopo do microsserviço de validação de conformidade: sua função é produzir automaticamente o laudo de conformidade de cada arquivo perante a especificação de seu formato, em uma operação automatizável e independente de política institucional.

As decisões sobre como o fluxo deve reagir a um arquivo não conforme (rejeitar, migrar ou aceitar com ressalva, em função de formatos-alvo definidos) são decisões de política formal de preservação e não se encaixam nos objetivos deste trabalho, voltado a instituições de pequeno porte que frequentemente não terão políticas formais de preservação implementadas.

4.1.4 Microsserviços no modelo Dappert-Farquhar

O modelo Dappert-Farquhar, apresentado na seção 2.3.4, descreve ações que se relacionam diretamente à caracterização e monitoramento de propriedades ou características significativas de objetos digitais. A principal contribuição desse modelo na identificação de microsserviços é sua ênfase no monitoramento contínuo dos requisitos de preservação e na avaliação de riscos de obsolescência de formatos.

O modelo fundamenta explicitamente o microsserviço de registro de eventos de preservação. Conforme apontam Pavão, Caregnato e Rocha (2016) sobre o modelo Dappert-Farquhar, a execução de cada ação de preservação é entendida como evento, e os executores dessas ações são os agentes. Esse registro é indispensável para a comprovação de autenticidade e integridade do acervo ao longo do tempo.

4.1.5 Microsserviços no modelo UC3

O conceito de microsserviço aplicado à preservação digital é mais diretamente fundamentado pelo modelo UC3, pois foi nesse cenário institucional que Abrams *et al*. (2011) propuseram o desmembramento de cadeias de ações de curadoria em pequenas unidades funcionais independentes, onde cada unidade representa uma

ação e realiza uma função específica que pode ser combinada com outras para gerar fluxos mais complexos e adaptáveis, o que também caracteriza perfeitamente os microsserviços.

O modelo UC3 organiza os microsserviços em camadas granulares e independentes enfatizando a importância de ações para armazenamento com replicação, validação de integridade, geração e curadoria de metadados, caracterização e transformação para criação de representações derivadas. Essa abordagem trata a curadoria digital como um processo contínuo de curadoria (California Digital Library, 2010; Abrams; Kunze; Loy, 2010) e fundamenta a identificação de microsserviços equivalentes a essas ações.

Entre as funções de preservação do modelo, destaca-se ainda a Caracterização. Conforme o projeto JHOVE2, associado ao UC3, a caracterização não se restringe à identificação e extração de propriedades técnicas de um arquivo, mas utiliza essas informações para realizar a validação de formato, entendida como “a determinação do grau de conformidade de um objeto às exigências normativas da especificação autoritativa de seu formato” (Abrams; Morrissey; Cramer, 2009). Dessa forma, o modelo UC3 fundamenta não apenas o microsserviço de identificação de formato, mas também o de validação de conformidade de formato.

4.1.6 Microsserviços no modelo Pavão-Caregnato-Rocha

O modelo Pavão-Caregnato-Rocha (2016) é o que mais diretamente articula em um único conjunto conceitual os elementos dos demais modelos analisados nesta pesquisa. A contribuição desse modelo para a identificação de microsserviços essenciais se divide em dois aspectos complementares.

O primeiro aspecto é estrutural, na organização dos serviços de preservação digital em categorias sequenciais com origem no ciclo de vida do DCC e do modelo funcional do OAIS, distinguindo serviços de curadoria digital como criação ou recebimento, avaliação e seleção, transformação, conversão, acesso e reuso, dos serviços de preservação propriamente ditos, como ingestão, ações de preservação e ações de armazenamento (Pavão; Caregnato; Rocha, 2016). Essa categorização foi a base para estruturação do Quadro 2 apresentado na seção seguinte.

O segundo aspecto é operacional. Pavão, Caregnato e Rocha (2016) relacionam explicitamente os microsserviços identificados pela síntese dos modelos, organizados por categoria.

Os microsserviços são: de ingestão, a verificação de vírus e integridade do documento, validação do formato, verificação da integridade do pacote e notificação da aceitação ao fornecedor; de descrição, a extração e produção de metadados técnicos e o registro das ações de preservação realizadas; de armazenamento, dentre os quais pode-se citar a replicação de arquivos e substituição de mídia; de monitoramento da preservação, para verificar a integridade dos documentos; de acesso e uso, relativos a busca e a geração de pacotes de disseminação e por último os microsserviços de transformação que são a migração de formato e a geração de versões. (Pavão; Caregnato; Rocha, 2016, p. 421)

A leitura dessa enumeração mostra que Pavão, Caregnato e Rocha (2016) tratam "ingestão", "descrição", "armazenamento", "monitoramento", "acesso e uso" e "transformação" como categorias funcionais, e não como microsserviços individuais. Cada categoria abriga, segundo os próprios autores, microsserviços nomeados separadamente. Dentro da categoria de ingestão, por exemplo, os autores listam como microsserviços distintos "a verificação de vírus e integridade do documento, validação do formato, verificação da integridade do pacote e notificação da aceitação ao fornecedor".

Esse detalhe é importante para a estruturação adotada neste trabalho: a ingestão é compreendida aqui como uma categoria dentro da qual operam microsserviços autônomos como a validação de admissão (que realiza a varredura antivírus do material recebido) e o empacotamento (que constitui ou valida o pacote de informação), e não como um microsserviço único. Da mesma forma, a menção explícita à "validação do formato" como item distinto na enumeração corrobora a separação adotada neste trabalho entre o microsserviço de identificação de formato e o de validação de conformidade de formato.

Essa lista de microsserviços também representa um fonte de referência direta para a identificação e categorização dos microsserviços essenciais a partir dos microsserviços identificados nesta seção nos diversos modelos conceituais analisados, já que é a única entre esses modelos que nomeia e agrupa explicitamente os microsserviços por categoria funcional. Os demais modelos contribuem com embasamento conceitual para cada microsserviço que foi listado.

4.1.7 Microsserviços no projeto InterPARES

A contribuição do InterPARES, apresentado na seção 2.3.7, para a identificação de microsserviços essenciais não está na proposição de funções computacionais, mas na formalização dos requisitos de autenticidade e integridade que essas funções devem assegurar. Ao definir que um registro digital é autêntico quando mantém sua identidade e integridade ao longo de uma cadeia de custódia ininterrupta e documentada (Duranti, 2005), o InterPARES fundamenta diretamente dois microsserviços: a verificação de integridade, que comprova que o objeto permanece inalterado, e o registro de eventos de preservação, que documenta cada ação sobre o objeto sem deixar lacunas em sua história custodial. Dessa forma, o InterPARES não amplia a lista de microsserviços candidatos, mas reforça o caráter essencial de dois deles, ao vinculá-los explicitamente ao requisito de autenticidade.

4.1.8 Microsserviços no framework NDSA Levels of Digital Preservation

O framework NDSA Levels organiza as atividades de preservação em cinco categorias funcionais — armazenamento e integridade geográfica, integridade e autenticidade dos arquivos, controle de informações, metadados e conteúdo dos arquivos — distribuídas em quatro níveis progressivos de maturidade (NDSA Levels of Preservation Working Group, 2019).

Sua contribuição é particularmente relevante para este trabalho por duas razões. A primeira é que seus níveis 1 e 2 foram projetados para serem alcançáveis por qualquer instituição, independentemente de tamanho e orçamento, o que os torna uma referência direta para o patamar mínimo de preservação adequado às instituições de pequeno porte.

A segunda é que suas categorias funcionais corroboram a essencialidade de vários microsserviços já identificados nos demais modelos. A categoria de armazenamento fundamenta o microsserviço de armazenamento com replicação, ao recomendar múltiplas cópias em locais distintos já no nível mínimo. A categoria de integridade e autenticidade fundamenta a validação contra vírus, a verificação de integridade por checksums e o registro de eventos; e as categorias de metadados e

de conteúdo fundamentam a identificação de formato e a extração de metadados técnicos. A categoria de Conteúdo fundamenta a geração de representações derivadas, ao prever, em seus níveis mais altos, migrações e normalizações de formato para assegurar o acesso continuado ao conteúdo.

O framework reforça, ainda, que a sustentabilidade dessas práticas técnicas depende de decisões organizacionais em termos de periodicidade, responsabilidades e documentação, uma observação retomada na definição dos microsserviços contextuais.

4.2 MICROSSERVIÇOS ESSENCIAIS PARA UMA INSTITUIÇÃO DE MEMÓRIA DE PEQUENO PORTE

A identificação dos serviços de preservação digital que são essenciais para uma instituição de memória de pequeno porte é realizada por meio de duas etapas.

Na primeira etapa, os microsserviços de preservação digital identificados nos modelos OAIS, DCC, PLANETS, Dappert-Farquhar, UC3, Pavão-Caregnato-Rocha, InterPARES e NDSA Levels of Digital Preservation são sistematizados. Essa sistematização envolveu a identificação do microsserviço, a definição de um nome para cada microsserviço, assim como relação de cada microsserviço com os componentes funcionais de OAIS e de Pavão, Caregnato e Rocha (2016). Foi então verificada a ocorrência de cada microsserviço em cada modelo analisado.

O Quadro 2 traz a identificação e nomenclatura dada a cada microsserviço, assim como a apuração da sua recorrência nos oito modelos analisados na Seção 4.1. A coluna categoria reflete as estruturas propostas no modelo OAIS e por Pavão, Caregnato e Rocha (2016), que reuniram e sintetizaram essas categorias funcionais a partir dos diversos modelos conceituais analisados naquele trabalho. A coluna recorrência demonstra aqueles microsserviços que mais ocorrem nos modelos estudados.

Quadro 2 — Recorrência dos microsserviços candidatos nos modelos conceituais

Categoria	**Microsserviço candidato**	**OAIS**	**DCC**	**PLAN.**	**D-F**	**UC3**	**PCR**	**IntP**	**NDSA**	**Rec.**
Seleção	Avaliação e seleção		✓					✓		2/8

Categoria	Microsserviço candidato	OAIS	DCC	PLAN.	D-F	UC3	PCR	IntP	NDSA	Rec.
Ingestão	Validação de admissão (AV)	✓					✓		✓	3/8
Ingestão	Atribuição de identificador persistente	✓				✓				2/8
Ingestão/Armazenamento	Identificação de formato			✓	✓		✓		✓	4/8
Descrição/Caracterização	Validação de conformidade de formato		✓	✓		✓	✓			4/8
Descrição/Caracterização	Extração de metadados técnicos	✓	✓	✓		✓	✓		✓	6/8
Armazenamento	Verificação de integridade (fixity)	✓				✓	✓	✓	✓	5/8
Ingestão/Armazenamento	Empacotamento (SIP/AIP)	✓	✓		✓		✓			4/8
Descrição/Caracterização	Registro de eventos de preservação		✓		✓	✓	✓	✓	✓	6/8
Transformação	Geração de representações derivadas	✓	✓	✓		✓	✓		✓	6/8
Armazenamento	Armazenamento com replicação	✓	✓			✓	✓		✓	5/8
Armazenamento	Refrescamento/substituição de mídia	✓					✓			2/8
Armazenamento	Reparação de arquivos corrompidos					✓			✓	2/8
Monitoramento	Monitoramento de formatos/obsolescência	✓	✓	✓	✓		✓		✓	6/8
Monitoramento	Monitoramento da comunidade designada	✓	✓							2/8
Acesso e uso	Geração de pacote de disseminação (DIP)	✓	✓			✓	✓			4/8
Acesso e uso	Anotação/enriquecimento por consumidores					✓				1/8

Fonte: elaborado pelo autor, a partir da análise apresentada na Seção 4.1.
Legenda: PLAN. = PLANETS; D-F = Dappert-Farquhar; PCR = Pavão-Caregnato-Rocha; IntP = InterPARES; NDSA = NDSA Levels of digital preservation; Rec. = recorrência.

Conforme especificado na metodologia, as funções que alcançaram o limiar de três ocorrências constituem os microsserviços candidatos. Após identificar, nomear e identificar a recorrência, a segunda etapa envolve classificar cada microsserviço quanto aos requisitos de escopo e governança. Atendem ao requisito de escopo aqueles microsserviços que pertencem às categorias de preservação ingestão, preservação, transformação e armazenamento, excluídas as funções de seleção, acesso, uso e monitoramento; e atendem ao requisito de governança aqueles microsserviços que possuem a capacidade de operar no patamar mínimo de preservação responsável, sem pressupor ou depender da existência de política formal de preservação.

O Quadro 3 classifica cada microsserviço identificado no Quadro 2, relacionando os critérios de recorrência (Rec.), escopo (Esc.) e governança (Gov.) definidos no início desta seção.

Quadro 3 — Microsserviços de preservação digital para instituições de memória de pequeno porte

Nº	**Microsserviço**	**Função**	**Rec.**	**Esc.**	**Gov.**	**Classif.**
MS1	Validação de admissão (AV)	Verificar a presença de código malicioso nos arquivos recebidos e decidir sobre a admissibilidade do material no fluxo	3/8	✓	✓	Essencial
MS2	Identificação de formato	Identificar o formato de cada arquivo por assinatura binária e atribuir o PUID PRONOM	4/8	✓	✓	Essencial
MS3	Validação de conformidade de formato	Verificar se cada arquivo está bem-formado e conforme a especificação do formato declarado	4/8	✓	✓	Essencial

Nº	Microsserviço	Função	Rec.	Esc.	Gov.	Classif.
MS4	Extração de metadados técnicos	Extrair e registrar automaticamente as propriedades técnicas dos arquivos	6/8	✓	✓	Essencial
MS5	Verificação de integridade (fixity)	Gerar e verificar checksums para detectar corrupção de arquivos na ingestão e periodicamente no armazenamento	5/8	✓	✓	Essencial
MS6	Empacotamento do objeto digital (SIP/AIP)	Constituir o SIP a partir do material recebido e estruturar o AIP com representações e metadados segundo padrão definido	4/8	✓	✓	Essencial
MS7	Registro de eventos de preservação	Documentar cada ação sobre o objeto com agente, data e resultado, garantindo rastreabilidade	6/8	✓	✓	Essencial
MS8	Armazenamento com replicação	Guardar o AIP com cópias em locais físicos distintos para redundância e prevenção de perda de dados	5/8	✓	✓	Essencial
MS9	Geração de representações derivadas	Criar representações derivadas a partir da matriz preservada para normalização ou migração	6/8	✓	✓	Essencial
R1	Monitoramento de formatos e obsolescência	Verificar periodicamente se os formatos utilizados permanecem adequados	6/8	✗	✗	Contextual

Nº	Microsserviço	Função	Rec.	Esc.	Gov.	Classif.
		e se os objetos mantêm integridade no armazenamento				
R2	Geração de pacote de disseminação (DIP)	Criar o pacote de disseminação com representações e metadados adequados ao usuário e registrar a entrega	4/8	✗	—	Contextual

Fonte: elaborado pelo autor.
Legenda: Rec. = recorrência (Quadro 2, limiar ≥ 3/8); Esc. = escopo (ingestão, preservação, transformação e armazenamento); Gov. = governança (opera no patamar mínimo sem política formal de preservação). Classif. = Classificação; ✓ critério atendido; ✗ critério não atendido; — não aplicável. Essencial = microsserviço necessário ao workflow mínimo (atende recorrência, escopo e governança no cenário de pequeno porte); Contextual = microsserviço igualmente essencial ao ciclo de curadoria, mas com implementação diferida no workflow devido ao contexto institucional alvo deste trabalho.

Dos onze microsserviços candidatos (Quadro 3), nove foram classificados como essenciais e dois como contextuais. Os nove microsserviços essenciais constituem um conjunto de funções que serão usadas no workflow proposto nas seções seguintes. São eles:

a) validação de admissão (AV);
b) identificação de formato;
c) validação de conformidade de formato;
d) extração de metadados técnicos;
e) verificação de integridade (fixity);
f) empacotamento do objeto digital (SIP/AIP);
g) registro de eventos de preservação;
h) armazenamento com replicação;
i) geração de representações derivadas.

O microsserviço de validação de admissão é o ponto de partida do processamento automatizado. Sua função de barreira impede que material comprometido prossiga no workflow. Os microsserviços de identificação de formato, validação de conformidade de formato e a extração de metadados técnicos são pré-requisitos para qualquer decisão de preservação, pois, sem informação sobre as

características dos objetos, não é possível definir estratégias de armazenamento nem avaliar os resultados de transformações.

O microsserviço de verificação de integridade por checksums é o meio confiável de detectar alteração ou corrupção dos objetos e suas representações ao longo do tempo. O microsserviço de empacotamento padronizado constitui o SIP na entrada e estrutura o AIP, sendo condição para a interoperabilidade entre sistemas e para a transferência de custódia entre instituições.

O microsserviço de registro de eventos confere auditabilidade e rastreabilidade ao acervo, materializando o requisito de cadeia de custódia ininterrupta formalizado pelo InterPARES. O microsserviço de geração de representações derivadas assegura a disponibilidade desses formatos alternativos sem comprometer a integridade da matriz preservada.

Os dois microsserviços classificados como contextuais são igualmente essenciais ao ciclo de curadoria, mas permanecem fora do workflow por dependerem de um contexto institucional fora da caracterização das instituições de pequeno porte, por razões distintas e explícitas:

O microsserviço de monitoramento de formatos e obsolescência, embora de alta recorrência, falha no critério de escopo, pois o monitoramento é uma entidade fora do recorte de ingestão, preservação, transformação e armazenamento. Também falha no critério de governança, pois sua operação pressupõe uma política formal de preservação que defina periodicidade, critérios de risco para formatos e processos de transformação formalizados para atendê-los, decisões de governança tipicamente ausente nas instituições de pequeno porte.

O microsserviço de geração de pacote de disseminação (DIP) falha no critério de escopo, pois a entrega de dados estruturados ao usuário final pertence às categorias de acesso e uso, que extrapolam o recorte de ingestão, preservação, transformação e armazenamento adotado neste trabalho.

Esses microsserviços são registrados para evidenciar funções do ciclo completo previsto pelo modelo OAIS e pelas categorias do modelo Pavão-Caregnato-Rocha, e para orientar a evolução institucional futura, mas não são implementados no workflow.

A sequência lógica de execução dos microsserviços essenciais em um fluxo de trabalho apoiado em ferramentas de software livre na plataforma Linux constitui o objetivo central desta pesquisa, e é apresentada nas seções subsequentes.

5 APLICATIVOS DE SOFTWARE LIVRE QUE REALIZAM MICROSSERVIÇOS DE PRESERVAÇÃO DIGITAL

A seção 4 identificou microsserviços essenciais para serem usados em instituições de memória de pequeno porte. Esta seção analisa aplicativos ou bibliotecas de Linux que podem ser usados na implementação desses microsserviços.

A análise e seleção de aplicativos ou bibliotecas para implementar cada microsserviço essencial seguiu um conjunto de seis critérios (Quadro 4), derivados diretamente dos cinco aspectos que caracterizam a instituição de memória de pequeno porte na Seção 3.1.

Quadro 4 — Critérios de seleção dos aplicativos

Critério	Descrição
C1	Licenciamento livre ou de código aberto, sem custo de aquisição, para atender ao aspecto financeiro-orçamentário (Seção 3.1)
C2	Baixo consumo de recursos e execução em hardware comum, para atender ao aspecto de infraestrutura tecnológica (Seção 3.1);
C3	Instalação simples por gerenciador de pacotes e dependências mínimas, para atender aos aspectos de infraestrutura e de recursos humanos (Seção 3.1);
C4	Operação automatizável por linha de comando e integrável a scripts Python, para atender ao aspecto de recursos humanos (Seção 3.1), dispensando operadores especialistas na ferramenta
C5	Adequação funcional ao microsserviço, verificada em documentação técnica da ferramenta e testes práticos em ambiente Linux, para garantir que a ferramenta de fato realize a função;
C6	Maturidade e manutenção ativa, com adoção consolidada em atividades de preservação digital e manutenção ativa do projeto que implementa a ferramenta, para atender à necessidade de sustentabilidade de longo prazo apontada na Seção 2 e pelos NDSA Levels.

Fonte: elaborado pelo autor

Na análise e seleção, quando mais de um aplicativo ou biblioteca satisfaz os seis critérios de forma equivalente, adota-se como solução para implementação do microsserviço aquele de menor número de dependências, e registram-se os demais como alternativas documentadas, a serem empregadas conforme as condições específicas da instituição. Na análise, quadros comparativos são apresentados para cada microsserviço, sintetizando a capacidade de atendimentos de cada aplicativo ou

biblioteca a cada critério que foi estabelecido para a avaliação, assim como uma decisão de escolha e uma justificativa dessa decisão.

Além dos critérios estabelecidos, para cada microsserviço identificado como essencial na seção 4 serão observados:

a) os aplicativos ou utilitários selecionados e sua função no contexto do microsserviço;
b) o modelo de licenciamento sob o qual esse aplicativo ou utilitário é oferecido;
c) requisitos especiais de infraestrutura computacional para sua operação, se existentes.

A análise também considera que todos aplicativos ou bibliotecas são nativos ou com suporte ativo corrente na plataforma Linux, instaláveis pelos gerenciadores de pacotes nativos de distribuições Linux populares, sem necessidade de licenciamento proprietário.

As seções a seguir apresentam a análise dos aplicativos ou bibliotecas para cada microsserviço. A seção final 5.10 reúne e sintetiza as escolhas.

5.1 MICROSSERVIÇO DE VALIDAÇÃO DE ADMISSÃO (AV)

No contexto de instituições de memória de pequeno porte, o material que chega para preservação normalmente não vem acompanhado de garantias sobre sua segurança ou procedência técnica. Antes de qualquer outra operação sobre o objeto digital, é necessário verificar que os arquivos recebidos não contenham conteúdo malicioso que possa comprometer o repositório, os sistemas da instituição ou a integridade de objetos já preservados.

Essa verificação de admissibilidade é a primeira função da categoria de ingestão e, conforme estabelecido pelo modelo OAIS (CCSDS, 2012), integra as atividades de quality assurance que a entidade de Ingestão executa sobre o conteúdo submetido antes de aceitá-lo formalmente. Pavão, Caregnato e Rocha (2016, p. 421) nomeiam explicitamente "a verificação de vírus" como um dos microsserviços de ingestão, o que fundamenta seu tratamento, neste trabalho, como microsserviço autônomo, distinto do empacotamento. A função deste microsserviço é verificar os arquivos recebidos em busca de ameaças e decidir, com base no resultado, se o material está apto a prosseguir no fluxo de preservação. O quadro 5 elenca aplicativos

ou bibliotecas de Linux capazes de implementar o microsserviço de validação de admissão; relaciona o atendimento dos mesmos aos critérios estabelecidos; e apresenta e justifica suas escolhas para implementar o microsserviço.

Quadro 5 — Candidatos para a varredura antivírus

Candidato	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**Decisão e justificativa**
ClamAV (clamscan)	✓	✓	✓	✓	✓	✓	Adotado — atende C1–C6; único antivírus livre maduro, multiplataforma e com base de assinaturas atualizada continuamente.
rkhunter / chkrootkit	✓	✓	✓	✓	✗	✓	Descartado — falha C5: detectam rootkits do sistema, não realizam varredura de vírus em arquivos de acervo.
LMD (Linux Malware Detect)	✓	✓	✓	✓	✗	✓	Descartado — falha C5: ferramenta focada em detecção de malware, não detecta vírus.
Antivírus em nuvem (API)	✗	✓	✗	✓	✓	✓	Descartado — falha C1 (custo de assinatura) e C3 (dependência de serviço externo e conta).
Antivírus Comercial	✗	✓	✗	✓	✓	✓	Descartado — falha C1 (licenciamento) e C3 (instaladores proprietários).

Fonte: elaborado pelo autor, com base em Cisco Systems (2024).

Para a varredura antivírus, o aplicativo selecionado é o ClamAV, um antivírus de código aberto licenciado sob a GNU General Public License (GPL), muito utilizado em ambientes Linux em servidores de E-mail, servidores de arquivos e como proteção de estações de trabalho Linux. O ClamAV executa a varredura de arquivos através da comparação de assinaturas de vírus armazenadas em um banco de dados que é atualizado regularmente pela Cisco Systems, através da Cisco Talos (divisão de inteligência de ameaças da Cisco), e pode ser utilizado de maneira interativa ou integrado a scripts de automação.

A ferramenta de linha de comando clamscan possibilita a integração direta com pipelines de processamento, fornecendo códigos e mensagens de retorno padronizados que simplificam o gerenciamento de fluxo no workflow, onde o resultado da varredura alimentará um ponto de decisão que determina se o material segue para as próximas etapas ou é rejeitado, conforme será detalhado na Seção 6. O ClamAV consome poucos recursos computacionais, e sua instalação pode ser feita por meio do gerenciador de pacotes, sem exigir dependências complexas (Cisco Systems, 2024).

5.2 MICROSSERVIÇO DE IDENTIFICAÇÃO DE FORMATO

O microsserviço de identificação de formato tem o objetivo de identificar de forma precisa e automatizada o formato de cada arquivo recebido para preservação. Conforme estabelecido no projeto PLANETS, a identificação de formato é etapa inicial no processo de caracterização e também um pré-requisito para qualquer estratégia de preservação. Sem uma identificação precisa do formato, não é possível determinar ações de preservação apropriadas, ou avaliar a qualidade de futuras transformações (Farquhar; Hockx-Yu, 2007).

A identificação de formato é feita pela comparação das características binárias de cada arquivo com os registros de assinaturas de formatos do PRONOM, um registro técnico online e base de dados mantida pelos Arquivos Nacionais do Reino Unido (National Archives, 2023). É importante distinguir a identificação de formato da validação de conformidade. A identificação de formato responde à pergunta "qual é o formato deste arquivo?", enquanto a validação de conformidade é uma operação distinta que verifica a qualidade interna do formato, e é realizada por outro microsserviço. O quadro 6 elenca, avalia, seleciona e justifica escolhas de aplicativos ou bibliotecas de Linux capazes de implementar o microsserviço de identificação de formato. O aplicativo escolhido para este microsserviço é o DROID (Digital Record Object Identification), uma ferramenta de código aberto criada pelos Arquivos Nacionais do Reino Unido e disponível sob a licença BSD;

Quadro 6 — Candidatos para identificação de formato

Candidato	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**Decisão e justificativa**
DROID	✓	✓	△	✓	✓	✓	Adotado (padrão) — atende C1, C2, C4, C5, C6; C3 com restrição (exige JRE/OpenJDK). Mantida como padrão por ser a ferramenta oficial dos National Archives e referência institucional do registro PRONOM.
Siegfried	✓	✓	✓	✓	✓	✓	Alternativa (recomendada para grandes volumes) — atende C1–C6 plenamente, inclusive C3 (binário estático Go, sem Java), com melhor desempenho; adotar quando a dependência de Java for indesejável ou o volume for elevado.
File Utility (libmagic)	✓	✓	✓	✓	✗	✓	Descartado - falha C5, identifica somente tipos MIME gerais, não usa base PRONOM
FIDO	✓	✓	✓	✓	△	✗	Descartado — falha C6 (manutenção irregular) e C5 com restrição (base PRONOM menos atualizada).

Fonte: elaborado pelo autor, com base em Lehane (2014) e National Archives (2023).

O DROID identifica formatos utilizando assinaturas binárias extraídas do registro PRONOM, atribuindo a cada arquivo analisado o seu respectivo PUID — o identificador único de formato do PRONOM — além do formato real, versão do formato e MIME type. O aplicativo pode ser executado por meio de linha de comando usando um arquivo JAR, o que permite integração simples ao workflow que será definido na próxima seção, e gera relatórios em formatos CSV ou XML que simplificam a integração com outros microsserviços.

Para integrar o DROID em scripts Python, utiliza-se o módulo subprocess da biblioteca padrão. A base de assinaturas PRONOM é atualizada periodicamente e pode ser baixada ou atualizada pelo próprio aplicativo, via linha de comando. A dependência exigida pelo DROID é o Java Runtime Environment (JRE) na versão 8 ou superior, ou o OpenJDK na versão 17 ou superior, ambos normalmente instaláveis através dos gerenciadores de pacotes do Linux (National Archives, 2023).

Como alternativa de menor consumo de recursos, pode ser também selecionado o aplicativo Siegfried, uma ferramenta de linha de comando desenvolvida em linguagem Go e oferecida sob a licença Apache 2.0. Siegfried é uma ferramenta especialmente eficaz em ingestões que envolvem um grande volume de arquivos. É também compatível com a base de assinaturas PRONOM e gera saídas nos formatos YAML ou JSON. A ferramenta é disponibilizada como um binário estático, eliminando a necessidade de qualquer dependência ou ambiente de execução extra, e sua incorporação em scripts Python também é realizada por meio do subprocess. O Siegfried oferece um desempenho consideravelmente melhor do que o DROID ao processar grandes volumes de arquivos, tornando-se uma alternativa excelente para uso em acervos com grande volume de arquivos (Lehane, 2014).

5.3 MICROSSERVIÇO DE VALIDAÇÃO DE CONFORMIDADE DE FORMATO

O formato de um arquivo pode ser corretamente identificado e, ainda assim, estar truncado, corrompido ou em desacordo com a especificação do formato identificado, o que compromete sua preservação a longo prazo e sua capacidade de ser renderizado ou transformado por outras ferramentas.

O microsserviço de validação de conformidade de formato tem por função verificar se um arquivo está bem formado e é válido segundo a especificação do

formato declarado ou detectado, produzindo um parecer de conformidade que pode subsidiar decisões de preservação ou armazenado em registros de eventos. Essa função deriva das etapas de caracterização dos modelos UC3 e DCC, do componente de validação da characterisation no projeto PLANETS e é também definida por Pavão, Caregnato e Rocha (2016) como "validação do formato", item distinto da identificação na enumeração dos microsserviços de ingestão. O quadro 7 elenca, avalia, seleciona e justifica escolhas de aplicativos ou bibliotecas de Linux capazes de implementar o microsserviço de validação de conformidade de formato.

Quadro 7 — Candidatos para validação de conformidade de formato

Candidato	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**Decisão e justificativa**
JHOVE	✓	✓	△	✓	✓	✓	Adotado (padrão) — validador de conformidade de referência; cobre formatos comuns em acervos digitalizados (TIFF, JPEG, PDF, WAV, entre outros) com módulos específicos; C3 com restrição (Java, já presente por causa de DROID/FITS).
veraPDF	✓	✓	△	✓	✓	✓	Alternativa (complementar, PDF/A) — validador oficial da família PDF/A, mantido pela PDF Association; recomendado quando o acervo contém documentos em PDF/A; C3 com restrição (Java)..
MediaConch	✓	✓	✓	✓	△	✗	Alternativa (complementar, audiovisual) — validador de conformidade para formatos A/V (Matroska, FFV1, etc.); adotar quando o acervo for predominantemente audiovisual.
Jpylyzer	✓	✓	✓	✓	✓	✓	Alternativa (complementar, JPEG 2000) — Ferramenta especializada em formatos JPEG.

Fonte: elaborado pelo autor, com base em Open Preservation Foundation (2024), PDF Association (2024) e MediaArea (2024b).

O aplicativo selecionado é o JHOVE (JSTOR/Harvard Object Validation Environment), mantido pela Open Preservation Foundation e licenciado sob LGPL, por ser o validador de conformidade de maior adoção institucional em preservação digital.

O JHOVE opera por módulos específicos de formato, verificando se cada arquivo é well-formed e valid segundo a especificação correspondente, e produz um relatório (em texto ou XML) que pode ser registrado como metadado de caracterização e como evento de preservação em PREMIS.

Por já integrar o conjunto de ferramentas agregadas pelo FITS (aplicativo candidato a implementar o microsserviço verificação de integridade, analisado na

Seção 5.5), o JHOVE pode ser invocado tanto isoladamente, quando se deseja um parecer de conformidade explícito e independente, quanto por meio do FITS, quando a validação é apenas um subproduto da extração de metadados.

Para acervos com documentos em PDF/A, recomenda-se complementar o JHOVE com o veraPDF, validador oficial da família PDF/A. Para acervos audiovisuais, o MediaConch oferece validação de conformidade equivalente. Todas as ferramentas são automatizáveis por linha de comando e integráveis a scripts Python via subprocess; a única ressalva de infraestrutura é a dependência de ambiente Java para JHOVE e veraPDF, já presente no workflow por causa de DROID e FITS.

5.4 MICROSSERVIÇO DE EXTRAÇÃO DE METADADOS TÉCNICOS

O microsserviço de extração de metadados técnicos tem a função de extrair automaticamente as propriedades técnicas dos arquivos que compõem o objeto digital. Isso inclui características técnicas como formato, versão, dimensões, resolução, espaço de cor, codec, taxa de amostragem e outras características.

As informações estruturadas resultantes dessa extração são utilizadas tanto para avaliar estratégias de preservação quanto para verificar a qualidade das ações de preservação executadas. Como exposto no referencial teórico (seção 2.4.1), os metadados técnicos são fundamentais para a preservação digital, e os os checksums e as propriedades de formato são especialmente importantes para garantir a integridade e autenticidade dos objetos ao longo do tempo (Rocha, 2025b).

O modelo UC3 enfatiza que a obtenção de metadados técnicos deve ser realizada na ingestão e também após cada ação de transformação, com o objetivo de preservar o histórico das características do objeto digital ao longo do tempo (California Digital Library, 2010). O quadro 8 elenca, avalia, seleciona e justifica escolhas de aplicativos ou bibliotecas de Linux capazes de implementar o microsserviço de extração de metadados técnicos.

Quadro 8 — Candidatos para extração de metadados técnicos

Candidato	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**Decisão e justificativa**
FITS	✓	✓	△	✓	✓	✓	Adotado (padrão) — atende C1, C2, C4, C5, C6; C3 com restrição (Java, já presente por causa do DROID). Agrega DROID, ExifTool, MediaInfo,

Candidato	C1	C2	C3	C4	C5	C6	Decisão e justificativa
							JHOVE e Tika e consolida tudo em um único XML, detectando conflitos.
ExifTool + MediaInfo + pyexiftool + pymediainfo	✓	✓	✓	✓	✓	✓	Alternativa (leve) — atende C1–C6 plenamente, sem Java; cobre os formatos mais comuns (imagens/documentos e A/V). Adotar quando não se deseja manter ambiente Java.
FFprobe (FFmpeg)	✓	✓	✓	✓	△	✓	Alternativa — atende os critérios com restrição em C5 (formato da saída), e os formatos são atendidos por FITS e MediaInfo.
JHOVE (isolado)	✓	✓	△	✓	✓	✓	Descartado (isolado) — Não utilizado aqui como extrator. Seu foco é a validação de conformidade, função tratada como microsserviço próprio na Seção 5.3. Integra o FITS, que o aciona internamente para fins de caracterização.
Metadata Extraction Tool (NLNZ)	✓	✓	✓	✓	✓	✓	Descartado — Falha C6 (sem manutenção): ferramenta histórica, legada, não é mais mantida.
Apache Tika (isolado)	✓	✓	△	✓	△	✓	Descartado (isolado) — C5 com restrição neste cenário (voltada a extração de texto); já incluída no FITS, pouco vantajosa sozinha.

Fonte: elaborado pelo autor, com base em Harvard Library (2023), Harvey (2024), MediaArea (2024a), FFmpeg Project (2024) e NATIONAL LIBRARY OF NEW ZEALAND, 2007.

O aplicativo selecionado para este microsserviço é o FITS (File Information Tool Set). Essa ferramenta foi desenvolvida pela Harvard Library e é disponibilizada sob a licença LGPL.

O FITS é um agregador de diversas ferramentas de caracterização, executa internamente várias ferramentas especializadas, como DROID, Exiftool, MediaInfo, JHOVE e Tika, e consolida todas as informações obtidas com essas ferramentas em um único documento XML padronizado. Essa estratégia de agregação é particularmente útil, pois oferece a possibilidade de identificar conflitos entre os resultados de diversas ferramentas, indicando arquivos que podem exigir verificação ou atenção adicional no processo de extração. O FITS é executado por linha de comando através de um script shell (fits.sh) e pode ser integrado a scripts Python por meio do módulo subprocess. Assim como o DROID, o FITS é escrito em Java, e, portanto, tem como dependência a disponibilidade de um ambiente Java no sistema (Harvard Library, 2023).

O aplicativo ExifTool, criado por Phil Harvey e disponível sob licenciamento GPL, é a opção mais apropriada onde o requisito é leveza e rapidez de execução. É uma ferramenta de linha de comando que pode ler e escrever metadados em diversos

formatos de arquivo, gerando saídas em texto simples, JSON ou XML, e também oferece um wrapper Python (pyexiftool) para facilitar a integração sem a necessidade de subprocessos (Harvey, 2024).

Já o aplicativo MediaInfo, oferecido sob a licença BSD, oferece extração detalhada de metadados técnicos de áudio e vídeo para arquivos de mídia audiovisual, e dispõe de biblioteca Python nativa (pymediainfo). A combinação de ExifTool e MediaInfo é simples de instalar através dos gerenciadores de pacotes em distribuições Linux e da API Python nativa. A combinação é eficiente e consome poucos recursos computacionais, abrangendo os formatos de arquivo mais utilizados em acervos de instituições de memória de pequeno porte (Harvey, 2024; MediaArea, 2024a).

5.5 MICROSSERVIÇO DE VERIFICAÇÃO DE INTEGRIDADE - FIXIDEZ

O microsserviço de verificação de integridade tem a função de identificar modificações ou corrupção nos arquivos que constituem os AIPs armazenados, realizando comparações de novos checksums gerados em uma verificação com as informações dos manifestos já existentes ou criados durante etapas que envolvem o empacotamento, realizado por aplicativos como BagIt-Python.

Quando as matrizes são empacotadas pela primeira vez, a criação dos checksums acontece durante a construção do pacote, e estabelece checksums iniciais para verificações posteriores. É nesse microsserviço que checksums existentes em manifestos são efetivamente comparados com novos checksums para determinar se os arquivos continuam íntegros.

A verificação periódica de integridade dos arquivos é fundamental para a detecção de corrupção silenciosa de dados, um fenômeno em que os arquivos se deterioram ao longo do tempo sem que erros visíveis sejam detectados ou gerados pelo sistema de arquivos. Essa verificação é considerada um requisito fundamental pelo modelo OAIS (CCSDS, 2012), está presente em todos os níveis do NDSA Levels of Digital Preservation e também é enfatizada no modelo Pavão-Caregnato-Rocha (2016) como elemento chave para a preservação da integridade dos objetos digitais em repositórios. O quadro 9 elenca, avalia, seleciona e justifica escolhas de aplicativos ou bibliotecas de Linux capazes de implementar o microsserviço de verificação de integridade.

Quadro 9 — Candidatos para verificação de integridade

Candidato	C1	C2	C3	C4	C5	C6	Decisão e justificativa
sha256sum (Coreutils)	✓	✓	✓	✓	✓	✓	Adotado (padrão, por arquivo) — atende C1–C6; SHA-256 é o algoritmo padrão do BagIt (RFC 8493), nativo no Linux.
hashlib (Python)	✓	✓	✓	✓	✓	✓	Adotado (em scripts) — atende C1–C6; mesmo papel sem chamada externa (subprocess), nativo do Python.
hashdeep	✓	✓	✓	✓	✓	✓	Adotado (verificação periódica) — atende C1–C6; gera e compara manifestos de árvores de diretórios inteiras (AIPs), detectando novos/ausentes/alterados.
b2sum	✓	✓	✓	✓	✗	✓	Descartado (função) — C5 com restrição: não suporta SHA-256, preferível para preservação.
md5sum	✓	✓	✓	✓	△	✓	Alternativa (compatibilidade) — C5 com restrição: MD5 é adequado só para validar pacotes herdados; SHA-256 é preferível para preservação.
Fixity (AVP)	✗	✓	✓	✓	✓	✗	Descartado — falha em C1, a versão atualizada moveu-se para modelo comercial/pago. Falha em C6, a versão open source não é mais mantida.

Fonte: elaborado pelo autor, com base em Library of Congress (2022) e Kornblum (2013).

Para este microsserviço foram selecionadas as ferramentas nativas do sistema operacional Linux sha256sum e md5sum, componentes do pacote GNU Coreutils e distribuídas sob a licença GPL.

Essas ferramentas calculam e verificam checksums de arquivos individuais com consumo mínimo de recursos, e sua integração em scripts Python é bastante trivial, tanto via subprocess, quanto por meio do módulo hashlib da biblioteca padrão Python, que dispensa qualquer dependência externa.

Para geração dos checksums o algoritmo SHA-256 é o mais indicado para preservação digital, pois oferece resistência adequada a colisões para a detecção de corrupção em aplicações práticas. Ele é o algoritmo adotado como padrão pelo formato BagIt (Kunze *et al*., 2018), implementado pela biblioteca BagIt-Python (Library of Congress, 2022).

Esse microsserviço é aplicado tanto no momento em que o pacote é armazenado, quanto em etapas posteriores de verificações regulares. Para o segundo caso, o aplicativo hashdeep é um conjunto de ferramentas, também distribuídas sob a licença GPL, que oferece funcionalidades adicionais relevantes para gerar informações úteis para verificação de integridade de conjuntos de arquivos e

diretórios. Ele possibilita a geração de manifestos de checksum para árvores de diretórios inteiros, comparação de conjuntos de arquivos com manifestos gerados anteriormente e detecção de arquivos novos, ausentes ou alterados. O hashdeep é especialmente eficaz para as verificações regulares de integridade de pacotes de armazenamento (AIP), gerando relatórios que podem ser registrados como eventos de preservação no esquema PREMIS (Kornblum, 2013).

5.6 MICROSSERVIÇO DE EMPACOTAMENTO DO OBJETO DIGITAL (SIP/AIP)

O microsserviço de empacotamento do objeto digital tem por função estruturar o objeto digital e os metadados produzidos sobre seu conteúdo em um recipiente lógico padronizado, uma condição necessária para a interoperabilidade entre sistemas e para a gestão de longo prazo do acervo.

Como apresentado no referencial teórico (seção 2.5.3), o modelo OAIS define três tipos de Pacotes de Informação: SIP, AIP e DIP, e o projeto europeu E-ARK criou o Common Specification for Information Packages (CSIP), uma especificação baseada nos padrões METS e PREMIS onde é definida uma arquitetura comum para esses pacotes (DILCIS Board, 2021).

O CSIP representa uma solução completa e altamente interoperável para o empacotamento de objetos digitais. No entanto, devido ao seu alto nível de detalhamento e à rigidez de suas regras de formato e validação, sua implementação exige ferramentas especializadas e um conhecimento técnico avançado, o que frequentemente ultrapassa a capacidade operacional e os recursos disponíveis nas instituições de pequeno porte. Para este trabalho foram examinadas soluções mais simples em padrões de empacotamento. O quadro 10 elenca, avalia, seleciona e justifica escolhas de aplicativos ou bibliotecas de Linux capazes de implementar o microsserviço de empacotamento do objeto digital.

Quadro 10 — Candidatos para empacotamento do objeto digital

Candidato	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**Decisão e justificativa**
BagIt-Python	✓	✓	✓	✓	✓	✓	Adotado — atende C1–C6; implementação de referência do RFC 8493, autocontida, com API Python nativa (menor número de dependências); cria SIP e AIP com manifesto de integridade.

Candidato	C1	C2	C3	C4	C5	C6	Decisão e justificativa
bagit-java	✓	✓	△	✓	✓	✓	Alternativa — também cria pacotes BagIt; C3 com restrição (exige Java). Alternativa em ambiente Java já existente (por DROID/FITS).
Bagger	✓	✓	△	✗	✓	✓	Descartado — cria/valida BagIt por interface gráfica (Java): falha C4 (automação) e C3 com restrição.
Exactly (AVP)	✓	✓	△	✗	✓	✓	Descartado — empacota e transfere por interface gráfica (Java): falha C4 (automação) e C3 com restrição.
RODA-in	✓	✓	△	✗	✓	✓	Descartado — gera SIP conforme E-ARK/CSIP por interface gráfica (Java): falha C4 e C3; o padrão CSIP também excede o perfil de pequeno porte. Demonstra, porém, que existe ferramenta livre madura para CSIP.
DART / DART Runner (APTrust)	✓	✓	△	✓	✓	△	Descartado — cria pacotes (BagIt e outros) com CLI automatizável; C3 com restrição (runtime próprio). (C6) irregular.

Fonte: elaborado pelo autor, com base em Library of Congress (2022), DILCIS Board (2021) e Academic Preservation Trust (2026)

Como solução de empacotamento para este microsserviço foi escolhido o padrão BagIt (RFC 8493), pois ele oferece o núcleo funcional básico de empacotamento autodescritivo com manifesto de integridade e a menor complexidade possível em termos de implementação e manutenção. Essa é uma escolha deliberada para adequação ao cenário da instituição de pequeno porte: os metadados BagIt não possuem a expressividade estrutural do CSIP/METS, mas atende perfeitamente aos requisitos de preservação identificados na seção 4, é amplamente suportado por ferramentas e sistemas de preservação digital e pode ser progressivamente estendido à medida que a maturidade institucional avança.

O aplicativo selecionado é o BagIt-Python, implementação de referência para o padrão BagIt em linguagem Python desenvolvida pela Biblioteca do Congresso americano e distribuída sob a licença Creative Commons Zero (CC0). Sua API nativa em Python elimina a necessidade de subprocess e torna a integração bastante simples.

Na constituição do pacote SIP, quando o material recebido é um diretório bruto, o BagIt-Python percorre os arquivos, calcula os checksums de cada um e gera os manifestos de integridade e os metadados iniciais do pacote.

É importante destacar que essa geração de checksums na ingestão não constitui verificação de integridade, pois não há valor de referência prévio com o qual comparar: trata-se do estabelecimento da linha de base que tornará possíveis as verificações periódicas realizadas pelo microsserviço de fixidez (Seção 5.5).

Quando, em outro cenário de utilização, o workflow recebe um SIP já empacotado em BagIt, o BagIt-Python realiza sua validação, comparando os checksums dos arquivos presentes com os manifestos do pacote e identificando arquivos ausentes, não declarados ou divergentes. Na estruturação do AIP, o BagIt-Python reúne as matrizes de preservação, os relatórios de identificação de formato, os arquivos de metadados técnicos e o bag-info.txt de metadados administrativos, recalculando os checksums e atualizando os manifestos. A instalação é feita pela API nativa do Python, sem dependências externas complexas.

Para objetos digitais simples, como fotografias ou documentos, o pacote BagIt com metadados técnicos fornecidos pelo FITS é suficiente no cenário de instituições de pequeno porte.

5.7 MICROSSERVIÇO DE REGISTRO DE EVENTOS DE PRESERVAÇÃO

O microsserviço de registro de eventos tem como objetivo documentar todas as ações realizadas sobre o objeto digital, vinculando cada ação ao agente que a executou, a data em que ela ocorreu e o resultado dessa execução, para assegurar rastreabilidade e auditabilidade do acervo ao longo do tempo.

O modelo Dappert-Farquhar estabelece que toda ação de preservação deve ser documentada como um evento, o que se torna um requisito essencial para todas as outras ações do fluxo, e não uma atividade opcional (Dappert; Farquhar, 2009). O modelo DCC reforça essa posição ao determinar que a descrição e a representação da informação devem ser realizadas em todas as etapas do ciclo de curadoria digital (Higgins, 2008).

Conforme foi estabelecido no referencial teórico (seção 2.4.1), os metadados de preservação são informações fundamentais para os objetivos de preservação digital a longo prazo, pois documentam a história dos eventos custodiais do objeto, garantindo a autenticidade e integridade desse objeto ao longo do tempo (PREMIS Editorial Committee, 2015). O quadro 11 elenca, avalia, seleciona e justifica escolhas de aplicativos ou bibliotecas de Linux capazes de implementar o microsserviço de

registro de eventos usando o padrão PREMIS. A abordagem adotada para a criação de registros PREMIS é a produção de arquivos XML em conformidade ao schema PREMIS por meio de scripts Python com a biblioteca lxml, facilmente instalável pelo gerenciador de pacotes Python.

Quadro 11 — Candidatos para geração registro de eventos de preservação

Candidato	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**Decisão e justificativa**
Scripts Python + lxml	✓	✓	✓	✓	✓	✓	Adotada — atende C1–C6; gera PREMIS XML conforme o schema, com controle total, sem sistema dedicado, e integra-se ao pacote BagIt.
Bibliotecas PREMIS de terceiros	△	✓	✓	✓	✓	✗	Descartada — falha C6 (pouco mantidas) e C1 com restrição (licenças variáveis); úteis se amadurecerem.
Módulo PREMIS de sistemas completos	✓	✗	✗	✓	✓	✓	Descartada — falha C2 e C3: pressupõe servidor dedicado e a infraestrutura que o trabalho busca evitar.
METS + PREMIS (digiprov)	✓	✗	✓	✓	✗	✓	Descartada — Falha C2 e C5: alta complexidade de codificação e estruturação METS, tornando o uso isolado do PREMIS mais sustentável para o cenário.
W3C PROV	✓	✓	✓	✓	✗	✗	Descartada — Falha C5, focado em proveniência na Web; falha C6; adoção institucional na área é nula se comparado ao PREMIS.

Fonte: elaborado pelo autor, com base em (PREMIS Editorial Committee, 2015).

O padrão de metadados adotado para o registro de eventos para este microsserviço é o PREMIS (Preservation Metadata: Implementation Strategies), realizado através de scripts Python e da biblioteca lxml. PREMIS é um esquema de metadados de preservação que define entidades e relacionamentos para o registro de informações estruturadas sobre objetos, agentes, direitos e eventos de preservação (PREMIS Editorial Committee, 2015).

Cada evento de preservação registrado pelo microsserviço armazena no registro PREMIS as seguintes informações: tipo de evento (ingestão, verificação de integridade, identificação de formato, migração de formato, ou outro identificador de tipo); identificador do objeto relacionado ao evento; data e hora de ocorrência do evento; agente responsável (ferramenta de software com versão e responsável institucional); e resultado alcançado (sucesso ou falha, com detalhes do erro quando aplicável). Os arquivos PREMIS são gerados e incorporados ao pacote BagIt como

metadados do objeto, assegurando que a rastreabilidade do acervo esteja integrada ao próprio pacote, sem a necessidade de depender de sistemas externos.

5.8 MICROSSERVIÇO DE ARMAZENAMENTO COM REPLICAÇÃO

O microsserviço de armazenamento com replicação tem a função de guardar o pacote de arquivamento (AIP) em mais de uma cópia, em locais físicos distintos, de modo a prevenir a perda de dados por falha de mídia, erro humano ou sinistro.

Conforme estabelecido na Seção 4, este microsserviço foi classificado como essencial em todos os critérios, e é importante mencionar a importância de sustentar diretamente o nível mínimo de preservação responsável definido pelos NDSA Levels (NDSA Levels of Preservation Working Group, 2019), que recomendam, já em seu primeiro nível, a manutenção de múltiplas cópias em locais separados. O modelo OAIS (CCSDS, 2012) e o modelo Pavão-Caregnato-Rocha (2016) também situam a replicação e a substituição de mídia entre os serviços de armazenamento.

A estratégia adotada como referência é a regra 3-2-1, uma definição que aparece em diversas recomendações na literatura de preservação digital: Manter três cópias dos dados, em pelo menos dois tipos de mídia, sendo ao menos uma cópia em local geograficamente distinto. No cenário de instituições de pequeno porte, essa estratégia é implementada de forma incremental: o ponto de partida viável é a replicação para múltiplos destinos locais (um segundo disco interno e um disco externo ou NAS em ponto de montagem), com a possibilidade de posterior adição de um destino remoto de baixo custo. O quadro 12 elenca, avalia, seleciona e justifica escolhas de aplicativos ou bibliotecas de Linux capazes de implementar o microsserviço de armazenamento com replicação, observando a regra regra 3-2-1.

Quadro 12 — Candidatos para armazenamento com replicação

Candidato	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**Decisão e justificativa**
rsync	✓	✓	✓	✓	✓	✓	Adotada (cópias locais) — atende C1–C6; sincronização incremental nativa no Linux, com verificação na transferência, para destinos locais e em rede local.
rclone	✓	✓	✓	✓	✓	✓	Adotada (cópia remota) — atende C1–C6; interface unificada para dezenas de provedores de nuvem (Backblaze B2, Google Drive) com

Candidato	C1	C2	C3	C4	C5	C6	Decisão e justificativa
							verificação por hash; viabiliza o 3-2-1 a baixo custo.
sftp / scp	✓	✓	✓	✓	△	✓	Descartada — C5 com restrição: transferência ponto a ponto simples, sem sincronização incremental; útil para um destino remoto único via SSH. Protocolo pode ser usado via rclone.
SaaS de backup	✗	✓	△	✓	✓	✓	Descartada — falha C1 (custo de licença/assinatura).
Syncthing	✓	✓	✓	✗	✗	✓	Descartada — Falha C4 e C5: arquitetura espelhada (outros syncthings ativos nos destinos), não envia dados para armazenamentos em nuvem puros. Opera por monitoramento contínuo e não por gatilhos sob demanda por scripts Python

Fonte: elaborado pelo autor, com base em Free Software Foundation (2024), Rclone (2024).

Para as cópias locais e em rede local, a ferramenta selecionada é o rsync, utilitário de sincronização incremental distribuído sob a licença GPL, nativo nas distribuições Linux, que transfere apenas as diferenças entre origem e destino e permite verificação de integridade na transferência. Para a cópia remota que completa a regra 3-2-1, a ferramenta selecionada é o rclone, distribuído sob a licença MIT, que oferece interface unificada para dezenas de provedores de armazenamento em nuvem — incluindo opções de baixo custo como Backblaze B2 e contas de Google Drive — e realiza verificação por hash após a transferência. Ambas as ferramentas são instaláveis pelo gerenciador de pacotes, operam por linha de comando e integram-se a scripts Python por meio do módulo subprocess. A definição dos destinos das cópias e da periodicidade da replicação é uma decisão institucional, conforme observado pelos NDSA Levels quanto à necessidade de políticas organizacionais que sustentem as práticas técnicas.

5.9 MICROSSERVIÇO DE GERAÇÃO DE REPRESENTAÇÕES DERIVADAS

A função do microsserviço de geração de representações derivadas é produzi-las a partir da matriz de preservação do objeto digital, sem alterá-la ou modificá-la de nenhuma forma, e incorporá-las ao pacote como representações adicionais em outros formatos.

Como exposto na seção 2.5.1 do referencial teórico, o CONARQ (2010) estipula que o processo de digitalização deve gerar, no mínimo, dois tipos distintos de representação: a matriz, voltada à preservação de longo prazo com a máxima fidelidade ao original, e representações derivadas normalmente destinadas para funções de acesso ou uso nos modelos conceituais. No modelo OAIS, tanto a matriz quanto as representações derivadas a partir dela geradas podem integrar o AIP, que admite múltiplas representações de um mesmo objeto digital (CCSDS, 2012).

Os objetivos deste trabalho estão focados na preservação e não faz parte do escopo prover funções de acesso. Entretanto, é parte deste trabalho dispor de um microsserviço que gera representações dos objetos digitais para serem disponibilizadas por função de acesso.

Normalmente, os parâmetros técnicos utilizados na configuração de uma derivada de acesso, como a resolução de saída, perfil de compressão, codecs utilizados e profundidade de cor das imagens, não são escolhas aleatórias. Essas escolhas de configuração são determinadas a partir das características relevantes definidas no plano de preservação da instituição para cada tipo de objeto, conforme descrito no referencial teórico (seção 2.5.2).

Essas propriedades determinam quais características do objeto devem ser mantidas nas representações derivadas, variando de acordo com a natureza do objeto e as expectativas da comunidade de usuários (Dappert; Farquhar, 2009).

Em muitos casos, instituições de pequeno porte não possuem ainda um plano de preservação digital que especifique configurações para matrizes e derivadas. Nesse caso, uma alternativa inicial de planejamento da preservação seria adotar as recomendações de digitalização do CONARQ (2010). Essas recomendações oferecem um conjunto de parâmetros adequado para derivadas: resolução mínima de 300 dpi e formato PDF/A para documentos textuais e JPEG com qualidade mínima de 80% para imagens. Esses valores são adotados como padrão para produção de representações derivadas nesse microsserviço, sem depender de um plano de preservação formal. O quadro 12 elenca, avalia, seleciona e justifica escolhas de aplicativos ou bibliotecas de Linux capazes de implementar o microsserviço geração de representações derivadas. Estes aplicativos ou bibliotecas realizam a conversão de formatos de arquivos.

Quadro 13 — Candidatos para geração de derivadas

Candidato	C1	C2	C3	C4	C5	C6	Justificativa
ImageMagick (imagens)	✓	✓	✓	✓	✓	✓	Adotada (imagens) — atende C1–C6; centenas de formatos, integra via subprocess/Wand. GraphicsMagick e VIPS são opções leves equivalentes.
GraphicsMagick / VIPS	✓	✓	✓	✓	△	✓	Alternativa — Restrição em C5, pois não atende tantos formatos quanto ImageMagick.
LibreOffice headless	✓	✓	✓	✓	✓	✓	Adotada (documentos textuais) — atende C1–C6; gera o PDF/A recomendado pelo CONARQ a partir de grande variedade de formatos.
Ghostscript	✓	✓	✓	✗	✗	✓	Descartada — Falha em C4 e C5: ferramenta histórica; sintaxe CLI altamente complexa; não suporta a variedade de formatos suportados pelo LibreOffice headless.
Pandoc	✓	✓	✓	✓	✗	✓	Descartada – Falha C5, forte em formatos de markup, mas não suporta tantos formatos quanto LibreOffice.
FFmpeg (áudio/vídeo)	✓	✓	✓	✓	✓	✓	Adotada (áudio e vídeo) — atende C1–C6; cobre praticamente todos os codecs, rápido, com wrapper ffmpeg-python. HandBrake (CLI) é alternativa para transcodificação de vídeo.
HandBrake (CLI)	✓	✓	✓	✓	✗	✓	Descartada – Falha C5, não processa arquivos somente audio, em video o foco maior é compressão.

Fonte: elaborado pelo autor, com base em CONARQ (2010), ImageMagick Studio (2024), The Document Foundation (2024) e FFmpeg Project (2024).

O aplicativo escolhido para converter o tipo de arquivo mais comum em acervos de objetos digitalizados, as imagens rasterizadas, é o ImageMagick. O ImageMagick é um conjunto de ferramentas de processamento de imagens de código aberto, distribuído sob a licença Apache 2.0. As ferramentas são altamente eficientes do ponto de vista de consumo de recursos computacionais, são compatíveis com centenas de formatos de imagem e possibilitam a execução de conversões, redimensionamentos, ajustes de qualidade e criação de thumbnails por meio da linha de comando, o que as tornam facilmente integráveis em scripts Python, além de oferecer integração direta por meio de subprocess ou pela biblioteca Wand (ImageMagick Studio, 2024). O ImageMagick é facilmente instalável em sistemas Linux utilizando o gerenciador de pacotes nativo da distribuição. A biblioteca Wand também é facilmente instalável pelo gerenciador de pacotes Python.

Para conversão de documentos de texto foi selecionado o aplicativo LibreOffice. É um aplicativo distribuído sob a licença Mozilla Public License 2.0 que

permite a conversão de documentos de texto e apresentações para o formato PDF/A por meio de linha de comando em modo headless, sem interface gráfica, dessa forma podendo ser integrado diretamente em scripts Python. O formato PDF/A é recomendado pelo CONARQ (2010) para acesso a documentos textuais, e pode ser gerado diretamente pelo LibreOffice a partir de diversos formatos de origem. É um aplicativo instalável de forma simples pelos gerenciadores de pacotes das distribuições Linux, o modo headless consome poucos recursos computacionais e é bastante eficiente na conversão de arquivos textuais.

Para conversão de arquivos de áudio e vídeo, o aplicativo selecionado foi o FFmpeg. É uma ferramenta distribuída sob a licença LGPL que suporta praticamente todos os formatos de áudio e vídeo normalmente encontrados em arquivos digitais, é extremamente rápida e eficiente em consumo de recursos computacionais. Essa ferramenta permite a transcodificação e a geração de representações de acesso por meio da linha de comando, podendo ser integrado ao Python por meio do subprocess ou pelo wrapper ffmpeg-python. A ferramenta pode ser instalada de forma simples pelos gerenciadores de pacotes nativos das distribuições Linux, e o wrapper ffmpeg-python é facilmente instalável pelo gerenciador de pacotes Python.

5.10 SINTESE DOS APLICATIVOS SELECIONADOS

O quadro 14 apresenta uma síntese dos aplicativos selecionados para cada microsserviço, reunindo as informações apresentadas nas seções anteriores sobre a função de cada ferramenta no microsserviço associado, seus modelos de licenciamento e requisitos mínimos de infraestrutura computacional.

Todos os aplicativos relacionados no quadro 14 são compatíveis com as principais distribuições Linux, e são instaláveis sem profundos conhecimentos técnicos por meio de gerenciadores de pacotes nativos. Todos são distribuídos através de licenças não-proprietárias, sem necessidade de licenciamento proprietário, e não exigem grandes recursos computacionais em hardware para funcionar de forma eficiente. A única exceção ao padrão de licença não-proprietária é caso a instituição decida usar o ambiente Oracle JRE (Java Runtime Environment) ao invés do ambiente OpenJDK para suporte às ferramentas DROID e FITS.

A integração de todas as ferramentas ou aplicativos em scripts Python é possível utilizando o módulo subprocess da biblioteca padrão, ou com wrappers

Python também disponíveis através do gerenciador de pacotes Python. Muitas das soluções selecionadas, como BagIt-Python, pyexiftool, pymediainfo e hashlib fornecem APIs nativas em Python que não necessitam do subprocess para suas respectivas operações dentro de scripts Python. A próxima seção mostra a sequência lógica de execução desses microsserviços em um fluxo de trabalho estruturado.

Quadro 14 — Síntese dos aplicativos selecionados por microsserviço

Microsserviço	**Adotada (padrão)**	**Alternativa**	**Licença**	**Critério decisivo / ressalva**	**Requisitos**
Validação de admissão (5.1)	ClamAV	—	GPL	Único antivírus livre maduro (C1, C5, C6)	Linha de comando Linux; baixo consumo
Identificação de formato (5.2)	DROID	Siegfried	BSD / Apache 2.0	DROID é referência do PRONOM (C5, C6); ressalva C3 (Java, já exigido pelo FITS). Siegfried dispensa Java e é mais rápido em grandes volumes	JRE 8+ ou OpenJDK 17+ (DROID); binário Go (Siegfried)
Validação de conformidade (5.3)	JHOVE	veraPDF / MediaConch	LGPL / Livre	Validador de conformidade de referência (C5); ressalva C3 (Java); ferramentas com dependência Java (já presente por DROID/FITS)	JRE 8+ ou OpenJDK 17+
Extração de metadados técnicos (5.4)	FITS	ExifTool + MediaInfo + pyexiftool + pymediainfo	LGPL / GPL / BSD	FITS agrega e consolida em XML (C5); ressalva C3 (Java). Combinação leve atende C1–C6 sem Java	JRE 8+ ou OpenJDK 17+ (FITS); linha de comando Linux; baixo consumo
Verificação de integridade (fixidez) (5.5)	sha256sum / hashlib + hashdeep	md5sum	GPL / PSF	SHA-256 é padrão do BagIt (C5); todos nativos e sem dependências (C1–C3); hashdeep para árvores inteiras	Nativos no Linux/Python; consumo mínimo
Empacotamento (SIP/AIP) (5.6)	BagIt-Python	bagit-java / DART	CC0 LGPL	Menor complexidade (C3) com interoperabilidade	Python 3.6+;

Microsserviço	Adotada (padrão)	Alternativa	Licença	Critério decisivo / ressalva	Requisitos
				adequada (C5); CSIP/METS excede o perfil	
Registro de eventos de preservação (5.7)	Scripts Python + lxml (PREMIS XML)	—	PSF BSD	Sem sistema dedicado, controle total e integração ao BagIt (C2, C3, C4)	Python 3.6+ com lxml
Armazenamento com replicação (5.8)	rsync (local) + rclone (remoto)	sftp / scp	GPL / MIT	Ambos atendem C1–C6; viabilizam a regra 3-2-1 incremental a baixo custo	Instalável via apt; SSH, SMB, S3, Drive, Backblaze
Geração de representações derivadas (5.9)	ImageMagick / LibreOffice / FFmpeg	GraphicsMagick·VIPS / Ghostscript / HandBrake	Apache 2.0 / MPL 2.0 / LGPL	Cobertura por tipo de mídia (C5) com baixo consumo (C2); parâmetros conforme CONARQ (2010)	Instalável via apt; consumo de CPU variável conforme mídia

Fonte: elaborado pelo autor, com base em análise de documentação técnica e testes práticos em ambiente Linux

6 WORKFLOW PARA MICROSSERVIÇOS ESSENCIAIS DE PRESERVAÇÃO DIGITAL

Nesta seção é apresentado o fluxo de trabalho sugerido para a execução dos microsserviços essenciais de preservação digital em instituições de memória de pequeno porte, utilizando a plataforma Linux. O fluxo de trabalho organiza os microsserviços identificados na seção 4 e as ferramentas escolhidas para cada microsserviço na seção 5 em uma sequência lógica e encadeada de etapas, definindo para cada etapa os seus gatilhos de acionamento, as entradas esperadas e as saídas geradas, e documenta relações de dependência com outras etapas.

Como definido no percurso metodológico (Quadro 1), a proposta do workflow foi baseada na análise dos fluxos de preservação digital encontrados nos modelos conceituais do referencial teórico e sua lógica foi adaptada ao cenário das instituições de memória de pequeno porte, com suas limitações de disponibilidade de recursos computacionais e humanos. O resultado é um fluxo linear com um ramo periódico, que pode ser executado em um único servidor ou workstation Linux com baixo custo operacional, sem utilizar infraestrutura de contêineres, orquestradores ou sistemas de filas de processamento.

6.1 PRINCIPIOS DE ORGANIZAÇÃO DO WORKFLOW

O workflow está organizado a partir de três princípios complementares. O primeiro é o princípio de sequencialidade, com dependências explícitas. Cada etapa do fluxo de trabalho possui gatilhos de entrada e saídas definidas, e nenhuma fase começa sem que a anterior tenha sido concluída com sucesso em sua saída. Esse princípio garante a rastreabilidade do fluxo e evita que erros em etapas anteriores se repitam em etapas subsequentes.

O segundo princípio é o da atomicidade dos microsserviços, conforme proposto por Abrams, Kunze e Loy (2010), e adotado como base conceitual para as seções 4 e 5. Cada etapa do workflow é um microsserviço autônomo, realizado por uma ferramenta ou script particular, que pode ser substituído sem impactar outras fases. Esse princípio assegura que a operação do workflow seja flexível e possa ser ajustada a diferentes tipos de acervo e infraestrutura.

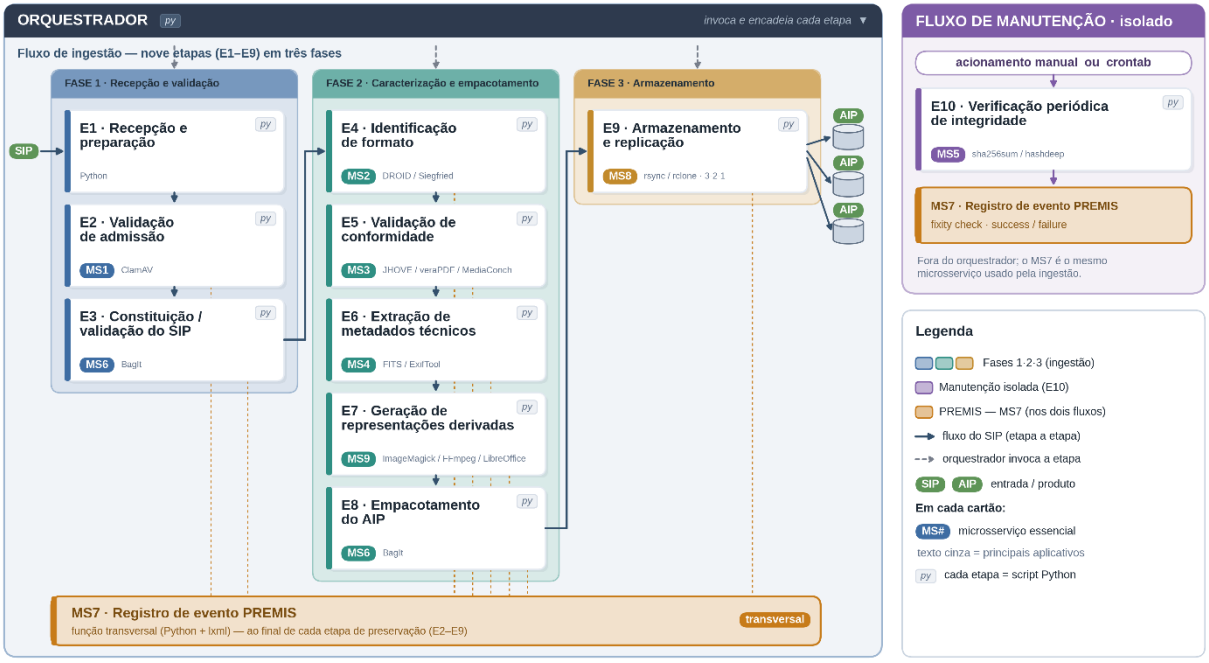
O terceiro princípio é o da parcimônia. O workflow só executa as etapas que são realmente necessárias para cada objeto digital, evitando processamento

redundante ou desnecessário. Falhas interrompem o workflow, e novas fases não são executadas em caso de falha. Ferramentas de transformação são aplicadas somente em objetos de tipos ou com características onde essa transformação é aplicável. Esse princípio é especialmente relevante para o cenário das instituições de preservação de pequeno porte, onde os recursos computacionais são limitados.

6.2 ESTRUTURA DO WORKFLOW

O workflow se divide em dois fluxos de natureza distinta: Um fluxo de ingestão, executado sobre objetos digitais a serem incorporados ao acervo, e um fluxo de manutenção, que é executado periodicamente sobre os pacotes AIP armazenados. Ambos compartilham microsserviços de validação, caracterização, de geração de registro de eventos PREMIS e de armazenamento e replicação e operam de forma independente e paralela (Figura 7).

Figura 7 — Visão geral da arquitetura do workflow



Fonte: Elaborada pelo autor (2026)

6.3 FASES E ETAPAS

O fluxo de ingestão é composto por nove etapas (E1 a E9), organizadas em três fases: a fase de recepção e validação (E1 a E3), a fase de caracterização e empacotamento (E4 a E8) e a fase de armazenamento (E9). O registro de eventos de preservação não constitui uma etapa, pois é função transversal, e o microsserviço que a implementa é executado ao final de cada etapa que realiza uma ação de preservação, gravando o respectivo evento PREMIS no pacote. O fluxo de manutenção possui uma única etapa periódica, a E10 (verificação de integridade), acionada por agendamento e executada de forma independente do fluxo de ingestão:

6.3.1 Fase de recepção, triagem e validação, etapas E1 a E3

A fase de recepção, triagem e validação compreende as etapas iniciais do workflow, responsáveis por receber e isolar o material submetido à preservação, diagnosticar seu formato e seu conteúdo, executar a varredura antivírus e normalizar qualquer entrada em um pacote BagIt válido, incluindo metadados quando estes acompanham o material de entrada. Seu objetivo é produzir um pacote de submissão padronizado e autossuficiente, independentemente da forma como o material foi entregue, em conformidade com a função de ingestão do modelo OAIS (CCSDS, 2012).

Em instituições de memória de pequeno porte não é incomum que os objetos digitais não estejam em um pacote de submissão (SIP) estruturado no momento da ingestão. Nesses casos, o início do fluxo de preservação parte de um conjunto de arquivos produzidos diretamente pelo processo de digitalização de objetos físicos ou produzidos diretamente por algum processo ou sistema digital, sem empacotamento, sem manifestos ou documentação de integridade e sem metadados. Essa situação determina que a fase de Recepção, triagem e validação não pode se limitar à verificação de um pacote já existente, ele deve também ser capaz de construir o pacote de submissão SIP a partir do material recebido. De acordo com o estabelecido pelo modelo OAIS (CCSDS, 2012), a existência do SIP não é obrigatória para a entidade funcional de ingestão. O que o modelo exige é que exista um acordo entre o produtor e o arquivo sobre o formato de entrega, o que esse workflow implementará aceitando vários formatos em suas etapas de recepção e triagem.

6.3.1.1 Etapa E1: Recepção e preparação do material

A etapa E1 é o ponto de entrada do fluxo e corresponde à função de recepção da submissão no modelo OAIS (CCSDS, 2012). As matrizes são depositadas em um diretório de entrada específico, onde o orquestrador, que é o componente do workflow que gerencia a execução do fluxo, procura por novos itens a serem ingeridos.

A etapa executa, de forma combinada, a recepção, a preparação e o diagnóstico. Os arquivos são movimentados para uma área de trabalho dedicada, sendo descompactados — caso tenham sido apresentados como arquivos compactados (ZIP, TAR ou TAR.GZ) — ou simplesmente copiados, quando já estão em formato de diretório aberto de matrizes ou pacote BagIt, preservando-se sempre o material original sem alterações no diretório de entrada.

Após essa movimentação, a área de trabalho é analisada para diagnóstico e verificação da presença de elementos como manifestos de checksum. Os produtos da E1 são uma cópia de trabalho isolada para as próximas etapas e um diagnóstico que será usado na etapa E3, para determinar como constituir ou validar o pacote SIP. Essa etapa não executa microsserviços de preservação, nem altera o conteúdo; ela é função de orquestração. Todo o processamento subsequente opera sobre a cópia, de modo que, em caso de falha ou rejeição, o original permanece disponível para reprocessamento.

6.3.1.2 Etapa E2: Validação de admissão (MS1)

A etapa E2 realiza a validação de admissão do material. É executada pelo microsserviço 1, de validação de admissão, conforme definido na Seção 5.1. Esse microsserviço executa o antivírus ClamAV sobre todos os arquivos do material recebido. O resultado da varredura é registrado em log e governa o primeiro ponto de decisão do fluxo: se forem detectados arquivos infectados, o material é movido para um diretório de quarentena, um evento de rejeição é registrado em formato PREMIS, e o fluxo é interrompido para aquele objeto, com notificação em mensagem de saída. Se a varredura não encontrar ameaças, o fluxo avança para a etapa E3.

6.3.1.3 Etapa E3: Constituição ou validação do SIP (MS6)

A etapa E3 realiza a constituição ou validação do SIP. É executada pelo microsserviço 6, de empacotamento do objeto digital, conforme definido na Seção 5.6.

Conforme o diagnóstico da etapa E1, a etapa E3 pode atuar de duas formas, dependendo da sofisticação do SIP. Na primeira, quando o SIP não contempla um pacote Bagit com checksum, ela empacota um diretório bruto de matrizes em um pacote BagIt, calculando os checksums e gerando os manifestos de integridade iniciais, que estabelecem a linha de base de fixidez. Na segunda, quando o SIP já contém um pacote BagIt, ela valida esse pacote recebido, gerando novos checksums e comparando os arquivos presentes com os manifestos já constantes no BagIt.

A saída da etapa E3 pode assumir duas formas: A primeira é um pacote BagIt válido, com fixidez estabelecida, metadados presentes na entrada e evento de preservação registrado em PREMIS. A segunda é um evento de rejeição, com notificação de causa em log.

O resultado da saída também determina o segundo ponto de decisão no workflow: Uma falha na constituição ou validação do SIP interrompe o fluxo, enquanto o sucesso avança o fluxo para a próxima etapa. Em ambos os casos, um evento de preservação em formato PREMIS é gerado.

6.3.2 Fase de caracterização e empacotamento, etapas E4 a E8

A fase de caracterização e empacotamento compreende as etapas que executarão microsserviços responsáveis por identificar e documentar características técnicas dos arquivos que integram o objeto digital, produzir metadados técnicos e reunir metadados produzidos na fase anterior para construir o Pacote de informação de arquivamento (AIP), que será submetido a fase posterior, de armazenamento.

6.3.2.1 Etapa E4: Identificação de formato (MS2)

A etapa E4 é a etapa de identificação de formato, executada pelo microsserviço 2, de identificação de formato, conforme definido na Seção 5.2. Esse microsserviço executa o DROID ou Siegfried sobre os arquivos do SIP, produzindo um relatório com o identificador único (PUID) PRONOM, a versão e o status de identificação (reconhecido ou não reconhecido) de cada arquivo.

O resultado é registrado e adicionado ao pacote como metadados de caracterização. Caso algum arquivo não seja reconhecido pelo registro PRONOM, o

microsserviço registra um alerta, sem interromper o workflow, pois a ausência de identificação é informação relevante para decisões de preservação, mas não impede a ingestão do objeto digital. A verificação de que o arquivo está em conformidade com a especificação do formato é distinta da identificação de formato, e é feita em outra etapa.

6.3.2.2 Etapa E5: Validação de conformidade de formato (MS3)

A etapa E5 é executada pelo microsserviço 3, de validação de conformidade de formato, e ocorre logo após a etapa de identificação de formato. A etapa verifica cada arquivo identificado pela etapa E4 para determinar se ele é bem formado e válido de acordo com a especificação do formato, e produz um parecer de conformidade.

A etapa executa o JHOVE sobre os arquivos do SIP, e pode ser complementada pela execução de outras ferramentas de validação formato, dependendo do formato do arquivo, como veraPDF para acervos com arquivos em PDF/A ou MediaConch para acervos audiovisuais.

O parecer de conformidade produzido é registrado como metadado de caracterização em evento PREMIS (microsserviço 7). Arquivos não conformes geram alerta em log e evento PREMIS, sem interromper o fluxo. A conduta diante de arquivos não conformes é uma decisão de política de preservação a cargo da instituição. Como este workflow é voltado para uso em instituições de pequeno porte, é adotada a política de somente documentar a conformidade ou não conformidade, sem suspender o fluxo.

6.3.2.3 Etapa E6: Extração de metadados técnicos (MS4)

A etapa E6, de Extração de metadados técnicos, é uma etapa realizada pelo microsserviço 4. Conforme descrito na seção 5.4, esse microsserviço executa as ferramentas de extração sobre os arquivos do SIP, conforme os formatos identificados na etapa E4, produzindo um documento XML que reúne metadados com as informações técnicas de cada arquivo. Esses metadados são armazenados para uso na etapa de empacotamento do AIP (E8). Como a extração deve ocorrer também após

transformações, o mesmo microsserviço é reexecutado sobre as representações derivadas geradas em E7.

Ao concluir, grava o evento PREMIS correspondente (microsserviço 7).

6.3.2.4 Etapa E7: Geração de representações derivadas (MS9)

A etapa E7 é a etapa de geração de representações derivadas, e é executada pelo microsserviço 9, de geração de representações derivadas (Seção 5.9).

O microsserviço opera sobre as matrizes de preservação do SIP, gerando representações derivadas conforme o plano de preservação da instituição ou, na ausência deste, os parâmetros do CONARQ (2010).

As representações derivadas são incorporadas ao próprio pacote, como representações adicionais à matriz, e serão consolidadas no AIP na etapa seguinte. O microsserviço 4 (extração de metadados técnicos) é reexecutado sobre as derivadas, documentando suas características.

Ao concluir, a etapa grava um evento PREMIS de transformação (microsserviço 7), com o tipo de transformação, os parâmetros, a ferramenta, a versão e o resultado. A constituição de um Pacote de Informação de Disseminação (DIP) a partir dessas derivadas não faz parte do fluxo essencial.

6.3.2.5 Etapa E8: Empacotamento do AIP (MS6)

A etapa E8 corresponde à função de empacotamento do AIP, executada pelo microsserviço 6, de empacotamento do objeto digital (Seção 5.6), o mesmo que constituiu o SIP na etapa E3.

A etapa reúne todos os elementos produzidos anteriormente — matrizes, representações derivadas, relatórios de identificação, pareceres de conformidade e metadados técnicos das matrizes e das derivadas — e os organiza em um pacote BagIt estruturado, como AIP.

Todos os checksums são recalculados e os manifestos atualizados, consolidando a linha de base de fixidez (microsserviço 5).

Ao concluir, gera um evento PREMIS de ingestão (microsserviço 7), consolidando o registro das ações do fluxo, inclui esse evento no PREMIS do AIP, e

novamente atualiza os checksums e manifestos para manter o pacote BagIt íntegro e válido.

6.3.3 Fase de armazenamento, etapa E9

A fase de armazenamento e disseminação compreende a etapa que executará o microsserviço responsável por guardar o AIP produzido na fase anterior em múltiplos destinos e registrar esses eventos de armazenamento em PREMIS.

6.3.3.1 Etapa E9: Armazenamento e replicação (MS8)

A etapa E9 compreende a função de Armazenamento e replicação, e utiliza o microsserviço 8, conforme definido na seção 5.8. Esse microsserviço implementa a transferência do pacote AIP finalizado na fase anterior para três destinos distintos, implementando a regra 3-2-1: um destino primário local e dois secundários, ao menos um geograficamente distinto. Após cada transferência, faz a verificação de integridade com base nos manifestos de checksum (microsserviço 5).

Por ocorrer somente após a finalização completa do AIP, a replicação garante que as três cópias sejam idênticas e consistentes, incluindo todas as matrizes, derivadas, e metadados produzidos até esta etapa, caso todas as transferências sejam bem sucedidas. Um evento PREMIS de armazenamento (microsserviço 7) registra cada destino e o resultado de cada transferência.

O resultado da etapa E9 também constitui o terceiro ponto de decisão do workflow.

6.3.4 Fase de Manutenção, etapa E10

A fase de verificação periódica de integridade compreende atividades que serão executadas periodicamente em um ciclo agendado para determinar o estado dos pacotes AIP armazenados em um diretório especificado.

6.3.4.1 Etapa E10: Verificação periódica de integridade (MS5)

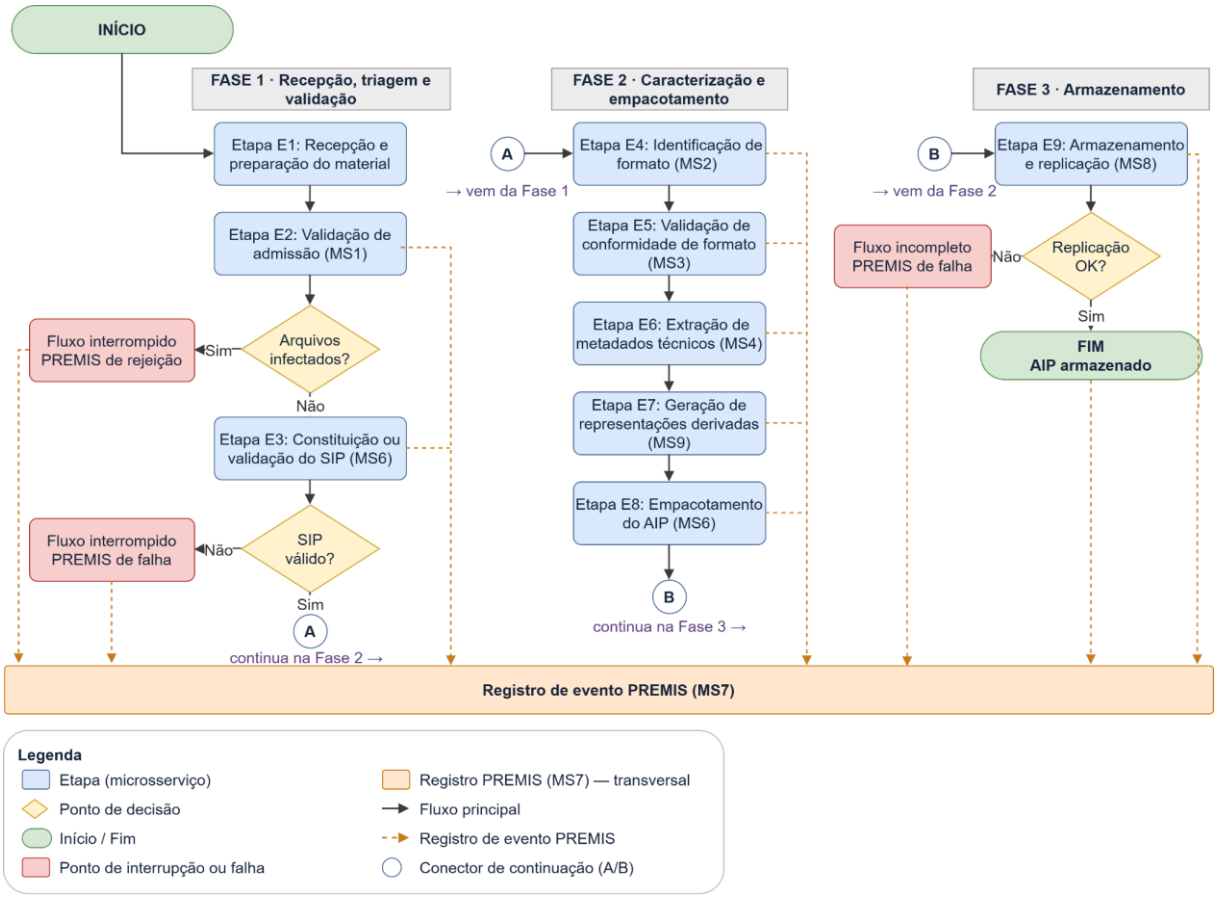
A etapa E10 é a etapa de verificação periódica de integridade, e é executada pelo microsserviço 5, de verificação de integridade, conforme descrito na seção 5.5. Essa etapa opera de forma independente fora do fluxo e das etapas descritas nas seções 6.3.1, 6.3.2 e 6.3.3. Esta etapa recebe um diretório ou ponto de montagem de armazenamento de AIPs, e processa todos os AIPs encontrados nesse local. Ela é acionada automaticamente pelo agendador cron do sistema Linux, em um ciclo repetitivo ou acionado manualmente quando necessário.

A verificação é feita pela geração de novos checksums, e comparação com os valores constantes nos manifestos nos pacotes armazenados. É gerado um evento de verificação de integridade em PREMIS XML para cada AIP verificado, com os resultados obtidos. Qualquer divergência gera notificação por mensagens de saída, alertando para a necessidade de restauração ou recuperação a partir de cópias ou de um dos destinos de replicação, e também um registro de failure em fixity check no registro PREMIS do AIP que apresentou o problema durante a verificação.

6.4 PONTOS DE DECISÃO EM FALHAS

O fluxo de trabalho define três pontos de decisão explícitos, onde o resultado de uma fase determina se o processamento do fluxo continua avançando ou se é interrompido. A Figura 8 apresenta o fluxo de execução das etapas E1 a E9. Nesta figura estes pontos de decisão são representados por losangos.

Figura 8 — Fluxograma das etapas E1 a E9



Fonte: Elaborada pelo autor (2026)

A primeira decisão acontece ao término da etapa E2, de validação de admissão (losango “Arquivos Infectados?” na Figura 8). Se forem identificados arquivos infectados por vírus, o objeto será movido para uma área de quarentena, será gerado um evento de rejeição PREMIS XML com a identificação dos arquivos afetados, e o fluxo será interrompido, com mensagens de saída de alerta.

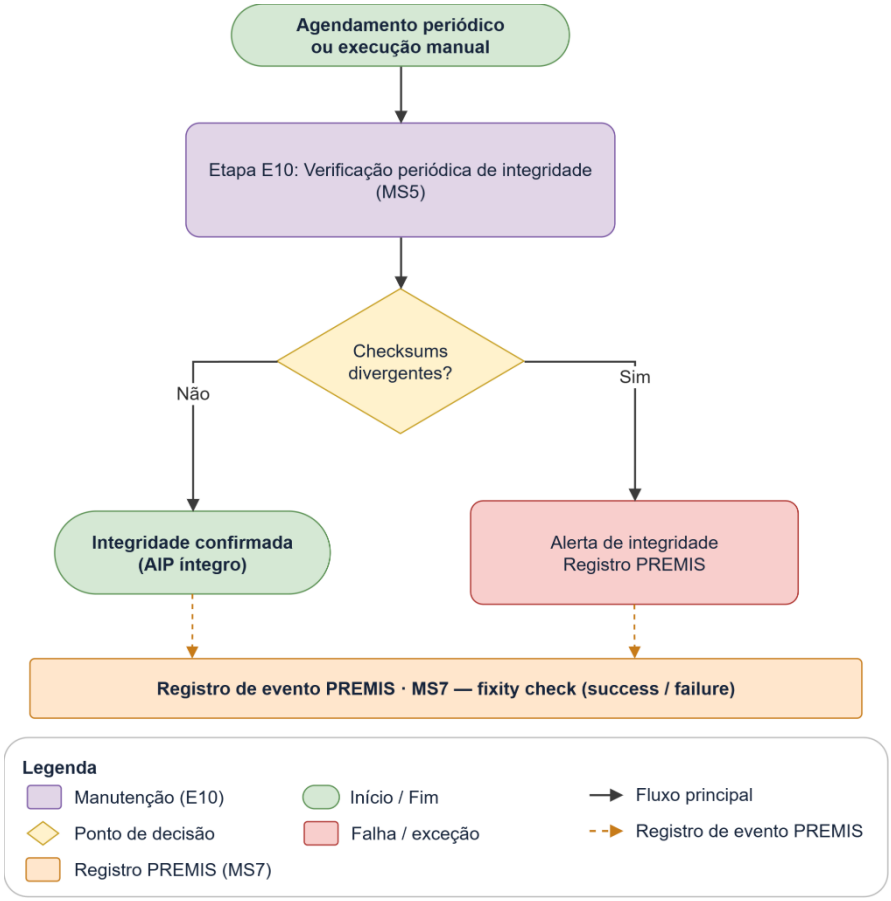
O segundo ponto de decisão ocorre ao final da etapa E3, de constituição ou validação do SIP (losango “SIP Válido?” na Figura 8). Se a validação do pacote falhar por divergência de checksum, arquivo ausente ou estrutura inválida, o fluxo é interrompido. Um evento PREMIS de falha de validação é gerado e uma mensagem de saída de alerta é exibida. O material permanece no diretório de entrada para reprocessamento.

O terceiro ponto de decisão ocorre ao final da etapa E9, de armazenamento e replicação (losango “Replicação OK?” na Figura 8). Se algum dos três destinos de replicação não puder ser alcançado ou se a verificação de integridade pós-

transferência falhar, um evento de armazenamento PREMIS registra a falha do destino específico e o workflow gera mensagem de alerta.

Em todos os pontos de decisão, a estratégia implementada é a falha explícita: o workflow interrompe o processamento e gera eventos e mensagens de alerta, para não continuar processando um objeto digital cujo estado é incerto. No caso do terceiro ponto de decisão, que envolve o microsserviço de replicação, a interrupção é relativa somente ao destino de replicação que falhou. Essa abordagem está alinhada com o princípio de auditabilidade que guia os modelos conceituais de preservação digital examinados no referencial teórico.

Figura 9 — Fluxograma da etapa E10, de verificação periódica de integridade (MS5)



Fonte: Elaborada pelo autor

Conforme podemos observar na Figura 9, dentro do fluxo da etapa 10, há também um ponto de decisão, que tem a única função de determinar o tipo de registro do resultado da verificação de integridade nos pacotes examinados. Uma falha de verificação registra os erros encontrados em log, e em um registro PREMIS de failure

de verificação de integridade. O sucesso faz o registro também em log, de sucesso, e em um registro PREMIS de success em evento de verificação de integridade.

6.5 QUADRO/SINTESE DAS FASES/ETAPAS

O Quadro 15 apresenta uma síntese das etapas do workflow, relacionando para cada etapa:

a) um código de identificação;
b) o nome;
c) o microsserviço correspondente conforme a seção 5;
d) as ferramentas utilizadas;
e) o gatilho ou entrada esperada e a saída produzida.

Quadro 15 — Etapas do workflow de microsserviços essenciais de preservação digital

Etapa	**Nome da etapa**	**Microsserviço**	**Ferramenta(s)**	**Gatilho / entrada**	**Saída**
E1	Recepção e preparação	—	Python	Diretório com arquivos bruto ou compactado, ou pacote SIP BagIt disponível no diretório de entrada	Cópia do material de entrada em área de trabalho isolada, com descompactação se necessário. Plano de processamento.
E2	Validação de admissão	MS1 (5.1)	ClamAV (clamscan)	Cópia de trabalho de E1	Relatório de varredura; Aprovação ou rejeição + evento PREMIS
E3	Constituição ou validação do SIP	MS6 (5.6)	BagIt-Python	Aprovação em E2	SIP BagIt válido + evento PREMIS

Etapa	Nome da etapa	Microsserviço	Ferramenta(s)	Gatilho / entrada	Saída
E4	Identificação de formato	MS2 (5.2)	DROID	SIP de E3	Relatório PUID/versão por arquivo + evento PREMIS
E5	Validação de conformidade de formato	MS3 (5.3)	JHOVE / veraPDF / MediaConch	Relatório da E4	Parecer de conformidade; alerta para não conformes + evento PREMIS
E6	Extração de metadados técnicos	MS4 (5.4)	FITS ou ExifTool + MediaInfo	Relatório da E4	Arquivo XML com metadados técnicos de cada arquivo + evento PREMIS
E7	Geração de representações derivadas	MS9 (5.9)	ImageMagick / LibreOffice / FFmpeg	Matrizes do SIP + metadados da E6	Derivadas incorporadas ao pacote + metadados das derivadas (MS4) + evento PREMIS de transformação
E8	Empacotamento AIP	MS6 (5.6)	BagIt-Python	Metadados técnicos da E4, E5, E6, SIP da E7	Pacote AIP BagIt com dados, metadados e manifestos + evento PREMIS de ingestão
E9	Armazenamento e replicação	MS8 (5.8)	SSH / sftp / rsync / rclone	AIP construído na E8	AIP replicado em destinos primário, secundário e remoto + evento PREMIS
E10	Verificação periódica de integridade	MS5 (5.5)	sha256sum / hashdeep	Agendamento periódico (cron)	Relatório de integridade por AIP; alerta em caso de divergência; evento PREMIS

Fonte: Elaborado pelo autor

A estrutura do fluxo de trabalho apresentada nesta seção estabelece a sequência lógica de encadeamentos e as responsabilidades de cada fase. Na seção 7, estão descritas a implementação dessa lógica em scripts Python executáveis na plataforma Linux, assim como os detalhes técnicos de cada um deles.

7 SCRIPTS PARA REALIZAR MICROSSERVIÇOS NO WORKFLOW

Esta seção descreve os scripts em Python que foram criados para automatizar a execução dos microsserviços essenciais no fluxo de trabalho estabelecido na seção 6. Como estabelecido no percurso metodológico (Quadro 1), a elaboração dos scripts irá implementar a lógica operacional do workflow em linguagem Python 3 na plataforma Linux. Os scripts estão disponíveis em repositório público com identificador persistente (Schmidt Júnior, 2026), cuja estrutura e correspondência com os microsserviços são detalhadas no Apêndice A.

7.1 ESTRUTURA / CONVENÇOES

O workflow é iniciado pela chamada do script orquestrador, detalhado na seção 7.2. Esta chamada ocorre através de uma linha de comando que recebe os parâmetros apresentados no quadro 16, que determinarão:

a) o diretório de entrada, onde ele buscará as matrizes ou pacotes para processamento;
b) o diretório de saída, onde ele criará o pacote AIP final;
c) um nome para identificação do trabalho ou sessão de processamento. O script criará um sufixo universalmente único que será agregado a esse nome. Caso o nome não seja informado, o nome do pacote AIP final será somente o identificador universal gerado;
d) opcionalmente, um nome de arquivo JSON que conterá configurações dos destinos de replicação para a Etapa E9.

Quadro 16 — Padrão de chamada do script orquestrador

```
python3 orquestrador.py \
    --entrada /caminho/entrada \
    --saida   /caminho/saida \
    --nome    lote_2026_03 \
    [ --destinos destinos.json ]
```

Fonte: Elaborado pelo autor

Os pacotes (SIP e AIP) processados durante a execução dos microsservicos são estruturados de acordo com o padrão de diretórios BagIt + PREMIS XML, com o

acréscimo de subdiretórios para metadados que não fazem parte dos metadados gerados pelo padrão do pacote BagIt, mas são gerados por outras etapas do workflow. Também será acrescido um subdiretório que receberá todos os logs de execução do orquestrador, com dados cronológicos e descrição de ocorrências. O padrão é apresentado no quadro 17 a seguir:

Quadro 17 — Padrão de diretórios criados pelo microsserviço de empacotamento (MS6)

```
nome-UUID/
├── bagit.txt
├── bag-info.txt
├── manifest-sha256.txt
├── tagmanifest-sha256.txt
└─ data/
│    ├── originais/
│    └── derivadas/
└─ metadata/
     └── premis.xml
└─ logs/
```

Fonte: Elaborado pelo autor

7.2 ORQUESTRADOR.PY

O orquestrador é o script que controla a execução do fluxo de ingestão, encadeando as chamadas aos demais scripts na sequência das etapas E1 a E9 e tomando decisões com base no código de retorno de cada microsserviço. Sua implementação segue o princípio da sequencialidade com dependências explícitas adotado na seção 6, isto é: nenhuma etapa é iniciada sem que a anterior tenha sido concluída com sucesso, ou com o registro de evento de erros, de modo que o objeto digital nunca avança no fluxo em um estado incerto.

Conforme estabelecido na Seção 7.1, o script recebe informações por linha de comando, por meio do módulo argparse da biblioteca padrão do Python: O diretório de entrada onde buscará o material, o diretório de saída onde o pacote será construído e um nome opcional para a sessão de processamento. A esse nome é agregado um prefixo “SIP” ou “AIP” conforme o ponto do workflow, e um sufixo universalmente único, gerado a partir do módulo uuid. Quando o nome não é informado, o identificador do pacote é apenas o próprio prefixo + sufixo. É aceito ainda um parâmetro opcional que indica o arquivo de configuração dos destinos de replicação utilizados na etapa E9.

Ao iniciar, o orquestrador cria a estrutura de diretórios da sessão e configura o registro de log para o subdiretório logs/ do pacote, gravando, em arquivo com registro de tempo e a descrição das ocorrências de toda a execução. A primeira etapa E1 consiste na recepção e preparação do material: o conteúdo do diretório de entrada é copiado para uma área de trabalho isolada — descompactando-se pacotes BagIt serializado, ZIP, TAR ou TAR.GZ quando necessário — preservando-se o original na entrada, e em seguida é feito um diagnóstico que identifica se o material recebido já constitui um pacote BagIt. Esse diagnóstico orienta o comportamento das etapas seguintes.

A partir da etapa E2, o orquestrador executa os scripts dos microsserviços dentro da ordem do workflow, avaliando o valor retornado por cada um. Foi adotada uma convenção de códigos de retorno padronizada, definida em um script auxiliar: o valor 0 indica sucesso e autoriza o avanço; o valor 1 indica falha que interrompe o fluxo; e o valor 2 indica um alerta que não configura uma condição fatal, que é documentada sem interromper o processamento. Essa convenção permite ao orquestrador aplicar a estratégia de tratamento de falhas descrita na Seção 6.4, dos pontos de decisão: a detecção de material infectado em E2 e a falha de constituição ou validação do SIP em E3 interrompem o fluxo; a impossibilidade de identificar um formato em E4 ou de validar a conformidade em E5 geram alerta e o fluxo prossegue; e a falha de um destino de replicação em E9 gera alerta relativo àquele destino, mas também não interrompe o fluxo. Concluída a etapa E8, o pacote de arquivamento (AIP) está consolidado no diretório de saída. E quando concluída a etapa E9, suas cópias estarão distribuídas conforme a política de replicação.

7.3 ADMISSAO.PY

O script de admissão implementa o microsserviço de validação de admissão (MS1) na etapa E2, descrito na Seção 5.1, e seu resultado corresponde ao primeiro ponto de decisão do workflow (Seção 6.4). Sua função é submeter todo o material recebido a uma varredura antivírus antes que ele seja admitido no repositório, impedindo que conteúdo malicioso seja incorporado ao acervo preservado.

A varredura é realizada pelo ClamAV, por meio da invocação do utilitário clamscan em modo recursivo sobre a cópia de trabalho produzida na etapa E1. A integração ocorre pelo módulo subprocess da biblioteca padrão, e a decisão de

admissibilidade é tomada a partir dos códigos de retorno padronizados que a ferramenta fornece: o código 0 indica que nenhuma ameaça foi encontrada e o material é aprovado; o código 1 indica a presença de arquivos infectados; e o código 2 indica erro durante a varredura. Quando arquivos infectados são detectados, o material é movido para um diretório de quarentena, um evento de rejeição é registrado no formato PREMIS e o fluxo é interrompido para aquele objeto. Quando ocorre erro de varredura, ou quando o utilitário não está disponível no sistema, o material também não é admitido, pois entende-se que material que não pôde ser verificado não pode ser admitido com segurança. Em todos os casos, o resultado é documentado em um evento PREMIS que registra a ferramenta empregada e sua versão, assegurando a rastreabilidade da decisão, e também produzidos registros no subdiretório logs/

7.4 EMPACOTAMENTO.PY

O script de empacotamento implementa o microsserviço de empacotamento do objeto digital (MS6), descrito na Seção 5.6, e é usado em dois momentos do workflow: na etapa E3, para constituir ou validar o pacote de submissão (SIP), e na etapa E8, para estruturar o pacote de arquivamento (AIP). Em ambos os casos, o empacotamento segue o padrão BagIt (RFC 8493), por meio da biblioteca de referência BagIt-Python, que dispensa chamadas a utilitários externos por oferecer interface nativa em Python.

Na etapa E3, o script opera em dois modos conforme o diagnóstico produzido na etapa E1. Quando o material recebido é um diretório bruto de matrizes, ele é organizado e empacotado em um SIP BagIt, com o cálculo dos checksums e a geração dos manifestos, estabelecendo-se a linha de base de fixidez do objeto. Quando o material já é recebido como um pacote BagIt, o script valida sua integridade, comparando os checksums dos arquivos com os valores declarados nos manifestos e detectando arquivos ausentes, não declarados ou divergentes. A falha nessa validação corresponde ao segundo ponto de decisão do fluxo (Seção 6.4) e interrompe o processamento, gerando registros de log e evento PREMIS, mantendo-se o material original na entrada.

Na etapa E8, o script consolida o AIP, reunindo as matrizes, as representações derivadas e os metadados produzidos pelas etapas anteriores, os checksums são recalculados e os manifestos atualizados, consolidando a fixidez do pacote final. Para

manter a estrutura definida na Seção 7.1, onde os subdiretórios metadata/ e logs/ estão no mesmo nível hierárquico do diretório de dados (data/), o script trata esses subdiretórios como diretórios de tag do BagIt, cobertos pelo manifesto de tags, de modo que a documentação de preservação e os registros de execução fiquem integrados ao pacote sem se confundirem com as matrizes preservadas.

7.5 IDENTIFICACAO.PY

Este script implementa o microsserviço de identificação de formato (MS2) e é executado na etapa E4, conforme a Seção 5.2. Sua função é determinar o formato de cada arquivo do pacote, ignorando a extensão do nome do arquivo e identificando-o por suas assinaturas binárias.

A identificação é feita com base no registro de formatos PRONOM, adotando-se como ferramenta padrão o DROID conforme a avaliação comparativa da Seção 5.2. A ferramenta é invocada pelo módulo subprocess e produz um relatório que associa a cada arquivo seu identificador PRONOM (PUID), o nome do formato e sua versão. Coerentemente com a Seção 6.4, a presença de arquivos cujo formato não pôde ser reconhecido não interrompe o fluxo e tais ocorrências são registradas como alerta em log e documentadas em evento PREMIS, cabendo à equipe da instituição decidir sobre o tratamento posterior. Dessa forma, a identificação cumpre seu papel informativo sem comprometer a continuidade da ingestão.

7.6 CONFORMIDADE.PY

O script de conformidade implementa o microsserviço de validação de conformidade de formato (MS3), é descrito na Seção 5.3 e é executado na etapa E5. A validação emprega ferramentas especializadas por tipo de formato: O JHOVE é usado como validador de referência, complementado pelo veraPDF para a família PDF/A e pelo MediaConch para formatos audiovisuais. Todas as ferramentas são invocadas pelo módulo subprocess do Python a partir do relatório de identificação produzido na etapa E4. O resultado é um parecer de conformidade que indica, para cada arquivo, se ele é bem formado e válido segundo a especificação do formato. Cabe salientar que essa é a função técnica do microsserviço, que é automatizável e

independente de política institucional, das decisões sobre como o fluxo deve reagir a um arquivo não conforme, que constituem política formal de preservação e não são implementadas devido ao foco do workflow em instituições de pequeno porte, que frequentemente não dispõem de políticas formais de preservação implementadas. O script adota a conduta de apenas documentar a conformidade ou a não conformidade. Caso ocorra a detecção de um arquivo não conforme é gerado alerta em log e registrado um evento PREMIS, sem interromper o fluxo (Seção 6.3.2.2). Desta forma o microsserviço cumpre sua função essencial de produzir o laudo de conformidade de cada arquivo sem interromper o fluxo de ingestão.

7.7 CARACTERIZACAO.PY

Este script implementa o microsserviço de caracterização, ou extração de metadados técnicos (MS4), conforme descrito na Seção 5.4, e é acionado em dois momentos: na etapa E6, sobre as matrizes de preservação, e na etapa E7, sobre as representações derivadas geradas pelo microsserviço MS9. Sua função é extrair e registrar as propriedades técnicas dos arquivos, como dimensões, resolução, profundidade de bits, codificação usada e demais características relevantes que constituem metadados essenciais para o monitoramento das propriedades significativas do objeto ao longo do tempo.

A extração utiliza como ferramenta padrão o FITS, que agrega e normaliza a saída de diversos identificadores e extratores, e, como alternativa documentada, a combinação de ExifTool e MediaInfo, conforme a avaliação da Seção 5.4. A ferramenta é invocada pelo módulo subprocess do Python, e os metadados extraídos são gravados em arquivos XML no subdiretório de metadados do pacote, distinguindo-se a caracterização das matrizes (E6) da caracterização das derivadas (E7) dentro de seus respectivos subdiretórios no diretório data.

7.8 DERIVADAS.PY

O script de geração de derivadas implementa o microsserviço de produção de representações derivadas (MS9), é descrito na seção 5.9 e executado na etapa E7. Sua função é produzir, a partir das matrizes de preservação, representações

derivadas em outros formatos, sem alterar ou modificar a matriz original de nenhuma forma. As derivadas são incorporadas ao próprio pacote, como representações adicionais da matriz, e consolidadas no AIP na etapa seguinte (na etapa E8). Essa separação entre a matriz e as representações dela derivadas assegura a disponibilidade de formatos alternativos sem comprometer a integridade da matriz preservada.

A ferramenta empregada varia conforme o tipo de arquivo, com base no relatório de identificação de formato produzido na etapa E4: imagens são convertidas com o ImageMagick, documentos textuais são convertidos para PDF/A por meio do LibreOffice em modo headless, e arquivos de áudio e vídeo são processados com o FFmpeg. Os parâmetros de conversão seguem as recomendações do CONARQ (2010), que estabelecem, entre outros, a resolução mínima de 300 dpi e o formato PDF/A para documentos textuais e o JPEG com qualidade mínima de 80% para imagens. As derivadas resultantes são incorporadas ao subdiretório próprio do pacote, e a transformação é registrada em evento PREMIS, documentando-se a ferramenta, os parâmetros aplicados e a relação entre cada derivada e sua matriz de origem.

7.9 EVENTOS.PY

O script eventos.py implementa o microsserviço de registro de eventos de preservação (MS7), descrito na Seção 5.7. Este microsserviço atua como uma função transversal no workflow, sendo acionado por praticamente todas as etapas do fluxo (E2 a E10): cada ação relevante executada sobre o objeto de preservação gera um evento que é documentado por este componente.

O registro é produzido no formato PREMIS, por meio da biblioteca lxml, assim não depende de outros sistemas, conforme a avaliação da Seção 5.7. Cada evento documenta o tipo de ação, a data e hora de ocorrência, o resultado alcançado e o agente responsável, distinguindo-se o agente de software do agente institucional. Os eventos são acumulados em um único documento XML, mantido no subdiretório de metadados do pacote. Quando o objeto já passou por etapas anteriores, o documento XML existente é carregado e os novos eventos são anexados a ele, preservando-se o histórico custodial ininterrupto exigido pelo InterPARES (2002). Com a integração desse registro ao pacote, a rastreabilidade das ações de preservação torna-se parte indissociável do objeto. O script é utilizado pelos demais como chamada de função,

mas também pode ser executado isoladamente para acrescentar um evento avulso a um pacote, se assim desejado pelo usuário.

7.10 REPLICACAO.PY

O script de replicação implementa o microsserviço de armazenamento com replicação (MS8), descrito na seção 5.8 e executado na etapa E9, e corresponde ao terceiro ponto de decisão do fluxo (Seção 6.4). Sua função é distribuir o AIP consolidado produzido na etapa E8 entre múltiplos destinos, materializando a regra 3-2-1 de redundância: três cópias, em pelo menos dois tipos de mídia, sendo ao menos uma em local externo.

A replicação é realizada por utilitários consolidados, invocados pelo módulo subprocess: o rsync, para o destino local e o destino em rede local, e o rclone, para o destino remoto, tipicamente em nuvem. Os destinos são definidos em um arquivo de configuração externo, o que permite à instituição adaptar a topologia de armazenamento sem alterar o código. Após cada transferência, o script verifica a integridade da cópia transferida, comparando-a com a origem, de modo a assegurar que a replicação não introduziu corrupção. O tratamento de falhas, conforme a Seção 6.4, é específico desta etapa: a indisponibilidade de um destino ou a falha na verificação pós-transferência não interrompe o processamento dos demais destinos, gerando-se um alerta relativo ao destino afetado e registrando-se o ocorrido em evento PREMIS. Assim, a falha de uma cópia não compromete a execução das demais nem o objeto já preservado.

7.11 VERIFICACAO.PY

O script de verificação implementa o microsserviço de verificação periódica de integridade (MS5), descrito na seção 5.5 e executado na etapa E10. Diferentemente das demais etapas, esta não integra o workflow normal, operando de forma independente e contínua ao longo do tempo: é acionada pelo agendador cron do sistema Linux, em ciclo definido pela política de preservação da instituição, para monitorar os pacotes já armazenados.

Para cada pacote AIP encontrado no diretório de armazenamento, o script recalcula os checksums dos arquivos e os compara com os valores constantes no manifesto BagIt do pacote, detectando divergências de checksum, arquivos declarados e ausentes e arquivos presentes mas não declarados. A verificação incide sobre a carga do pacote, nas matrizes e nas derivadas preservadas, que constituem o conteúdo cuja fixidez deve ser garantida. Os arquivos administrativos, como metadados e registros de eventos e de execução, crescem legitimamente ao longo da vida custodial do objeto, uma vez que cada nova verificação acrescenta um evento ao próprio histórico PREMIS, e por isso não são tomados como base da comparação de fixidez. O cálculo dos checksums emprega o módulo hashlib da biblioteca padrão, podendo o utilitário hashdeep ser empregado como reforço para árvores de diretórios inteiras. A execução de um ciclo produz um relatório de integridade por pacote e registra um evento PREMIS de verificação em cada pacote AIP. Na ocorrência de qualquer divergência é gerado um alerta em log recomendando a restauração do objeto a partir de uma das cópias de replicação produzidas na etapa E9. Fecha-se, assim, o ciclo de preservação do workflow, articulando a verificação de integridade à redundância de cópias como mecanismo de detecção e recuperação de perdas.

7.12 QUADRO SINTESE DOS SCRIPTS PARA REALIZAÇÃO DE MICROSSERVIÇOS

O Quadro 18 apresenta uma síntese dos scripts implementados para realizar os microsserviços das etapas do workflow, relacionando seus nomes, as etapas onde cado um é utilizado, sua função, as entradas esperadas e saída produzidas.

Quadro 18 — Scripts Python para o workflow de microsserviços de preservação digital

Script	Etapa	Função	Entradas	Saídas
orquestrador.py	E1-E9	Controla a execução do fluxo e encadeia chamadas aos scripts na sequência de etapas, registra mensagens de log, toma decisões com base no retorno dos scripts dos microsserviços.	Diretório de entrada Diretório de saída Identificador Arquivo de configuração de replicação (opcional)	Registro de logs e mensagens de saída.
admissao.py	E2	Executa a varredura antivírus e decide a admissibilidade	Cópia de trabalho	Log de varredura; Evento PREMIS
empacotamento.py	E3,E8	Constitui/valida o SIP (E3) e estrutura o AIP (E8) no formato BagIt	Material aprovado; matrizes + metadados técnicos + metadados conformidade	SIP/AIP BagIt válido com manifestos SHA-256 Evento PREMIS
identificacao.py	E4	Identifica o formato de cada arquivo do SIP por assinaturas binárias PRONOM	Caminho do SIP	Relatório CSV com PUID, versão e status de identificação de cada arquivo Evento PREMIS
conformidade.py	E5	Valida conformidade de formato (JHOVE/veraPDF); alerta para não conformes	Caminho do SIP; relatório de E4	Parecer de conformidade; evento PREMIS
caracterizacao.py	E6	Extrai metadados técnicos das matrizes e das derivadas	Caminho do SIP ou diretório de derivadas; relatório de identificação de formato (E4)	Arquivo XML com metadados técnicos de cada arquivo Evento PREMIS
derivadas.py	E7	Gera representações derivadas a partir das matrizes; Ferramenta conforme o tipo de arquivo	Caminho do SIP; relatório de identificação de formato (E4); parâmetros de conversão por tipo de arquivo padrão CONARQ (2010)	Representações derivadas (JPEG, PDF/A, MP3 etc.) incorporadas ao SIP Evento PREMIS

Script	Etapa	Função	Entradas	Saídas
eventos.py	E2, E3, E4, E5, E6, E7, E8, E9, E10	Gera e inclui arquivo PREMIS XML documentando os eventos de preservação;	Identificador do objeto; lista de eventos com agente, data e resultado	Arquivo PREMIS XML incluído no SIP; manifestos BagIt atualizados; Evento PREMIS
replicacao.py	E9	Replica o AIP finalizado na E8 para três destinos distintos: local (rsync), rede local (rsync/sftp) e remoto (rclone); verifica integridade pós-transferência	Caminho do AIP; configuração de destinos (rclone.conf; chaves SSH)	AIP replicado nos três destinos; log de replicação com resultado por destino; Evento PREMIS
verificacao.py	E10	Executa verificação periódica de integridade dos AIPs armazenados; compara checksums recalculados com manifestos BagIt	Diretório de armazenamento de AIPs	Relatório de integridade por AIP; evento PREMIS de verificação; alerta em caso de divergência

Fonte: elaborado pelo autor

8 AVALIAÇÃO

Esta seção apresenta a avaliação do workflow de microsserviços essenciais de preservação digital proposto na seção 6 e implementado através dos scripts descritos na seção 7. A avaliação atende ao objetivo específico desta pesquisa de avaliar o workflow em elementos de um acervo, e busca evidenciar se o conjunto de microsserviços, ferramentas de software livre e scripts é, de fato, utilizável nas condições de capacidade institucional, financeira e tecnológica reduzidas que caracterizam a instituição de memória de pequeno (seção 3.1), para responder à pergunta de pesquisa formulada na introdução.

8.1 NATUREZA DA AVALIAÇÃO

De acordo com a natureza qualitativa, aplicada e de caráter tecnológico desta pesquisa como explicitado na seção 3, a avaliação aqui apresentada não constitui um experimento controlado voltado à medidas estatísticas de desempenho, mas uma avaliação funcional com o propósito de demonstrar que o workflow executa as funções a que se propôs através de seus scripts e gera resultados aderentes aos requisitos de preservação digital identificados no referencial teórico. Como resultado de pesquisa aplicada de caráter tecnológico, esse artefato, o workflow e seus scripts, é o resultado da pesquisa, e a avaliação se ocupa de verificar a executabilidade e a validade dos produtos que ele gera.

A avaliação está organizada em torno de três aspectos de verificação: O primeiro aspecto é a executabilidade, onde será verificado se cada microsserviço, instrumentalizado pelas ferramentas de software livre selecionadas na seção 5 e implementado pelos scripts da seção 7, executa corretamente sobre os objetos digitais submetidos, em ambiente compatível com o de uma instituição de pequeno porte.

O segundo aspecto é a integração: Será verificado se o orquestrador encadeia as etapas E1 a E9 na sequência e com as dependências definidas na seção 6, e se os três pontos de decisão em falhas (Seção 6.4) operam conforme especificado.

O terceiro aspecto é a adequação aos requisitos de preservação, onde será verificado se o produto do workflow — os pacotes AIP válidos em formato BagIt, os metadados técnicos, os eventos em registro PREMIS, as representações derivadas e as cópias replicadas — atende aos requisitos de integridade, autenticidade,

rastreabilidade e redundância formalizados pelos modelos conceituais (OAIS, InterPARES, NDSA Levels) de onde os microsserviços foram derivados.

Adota-se como patamar de referência para o juízo de adequação o conjunto dos níveis 1 e 2 dos NDSA Levels of Digital Preservation, deliberadamente concebidos para serem alcançáveis por qualquer instituição independentemente de porte e orçamento (NDSA Levels of Preservation Working Group, 2019), e que correspondem ao “patamar mínimo de preservação responsável” que orienta o recorte de essencialidade desta pesquisa (Seção 4).

8.2 AMBIENTE DE AVALIAÇÃO

Para que a avaliação seja representativa das condições de uma instituição de memória de pequeno porte, o workflow foi executado em um ambiente que reproduz, em hardware virtualizado, uma configuração de computador de uso geral típica, com nível de recursos modesto, compatível com modelos domésticos ou básicos de escritório e sem licenciamentos proprietários nos softwares utilizados.

O ambiente foi constituído por uma máquina virtual provisionada no VirtualBox, executando o sistema operacional Ubuntu 26.04 LTS. As ferramentas de software livre foram instaladas a partir do gerenciador de pacotes nativo da distribuição (apt), e as dependências de Python foram isoladas em um ambiente virtual (venv), de modo a não interferir no Python do sistema e a manter a configuração reprodutível, uma prática adequada para um ambiente cuja expectativa de vida útil é de longo prazo. O quadro 19 sintetiza a configuração do ambiente, da máquina virtual e as versões das ferramentas efetivamente empregadas.

Quadro 19 — Ambiente de avaliação, máquina virtual e versões das ferramentas

Componente	**Especificação / versão**
Virtualização	VirtualBox 7.2.8
Recursos da máquina virtual usada para o workflow	4 vCPUs 8 Gb RAM 50 Gb disco virtual
Sistema operacional da máquina virtual	Ubuntu 26.04 LTS
Python	3.14.4
Antivírus (MS1)	ClamAV 1.4.4
Empacotamento (MS6)	BagIt-Python 1.9.0

Componente	Especificação / versão
Identificação de formato (MS2)	DROID 6.8.1 Siegfried 1.11.4
Conformidade de formato (MS3)	JHOVE Rel. 1.34.0, 2025-07-02 veraPDF 1.30.1 MediaConch 25.4
Metadados técnicos (MS4)	FITS 1.6.0 ExifTool 13.50 MediaInfo v26.01
Derivadas (MS9)	ImageMagick7.1.2-18 LibreOffice 26.2.2.2 FFmpeg 8.0.1
Integridade (MS5)	hashlib 3.14.4 (Python) hashdeep 4.4
Replicação (MS8)	rsync 3.4.1 rclone 1.60.1
Eventos (MS7)	lxml 6.1.1

Fonte: elaborado pelo autor.

Para a etapa de armazenamento com replicação (Etapa E9), foram configurados três destinos para implementar a regra 3-2-1: um destino primário local (disco da própria máquina), um destino secundário em rede local, e um destino remoto de baixo custo em nuvem acessado pelo rclone. A definição desses destinos é uma decisão institucional, e aqui foram fixados apenas para fins de demonstração do mecanismo.

8.3 CRITÉRIOS DE AVALIAÇÃO

A partir dos três aspectos de verificação, foram definidos critérios objetivos que permitem julgar, para cada execução, se o resultado obtido corresponde ao esperado. O quadro 20 apresenta esses critérios e o evidenciamento adotado para cada um.

Quadro 20 — Critérios de avaliação do workflow

Critério	Descrição	Como é evidenciado
CR1 — Executabilidade	Cada microsserviço executa sobre as entradas previstas sem erro fatal não tratado.	Registro em logs/ e código de retorno padronizado (0/1/2)

Critério	Descrição	Como é evidenciado
CR2 — Sequenciamento	O orquestrador encadeia as etapas E1 a E9 respeitando dependências; nenhuma etapa inicia sem conclusão da anterior.	Log de execução do orquestrador com a ordem das etapas
CR3 — Pontos de decisão	Os três pontos de decisão (Seção 6.4) interrompem ou prosseguem o fluxo conforme especificado	Comportamento observado no cenário da execução
CR4 — Validade do pacote	AIP é pacote BagIt válido; manifestos SHA-256 conferem com o conteúdo	Validação com bagit.py --validate; saída registrada em logs
CR5 — Completude dos metadados	PREMIS XML bem formado e com cadeia de eventos ininterrupta; metadados técnicos presentes	Inspeção do premis.xml no subdiretório metadata/
CR6 — Rastreabilidade	Todas as ações das etapas E2 a E10 geram evento PREMIS com tipo, data, agente e resultado	Inspeção do premis.xml no subdiretório metadata/
CR7 — Geração de derivadas	Representações derivadas produzidas conforme parâmetros do CONARQ (2010), sem alteração da matriz	Inspeção das derivadas e do evento PREMIS de transformação
CR8 — Redundância	AIP replicado em três destinos com verificação de integridade pós-transferência	Log de replicação por destino
CR9 — Viabilidade em hardware comum	Execução conclui em tempo e consumo de recursos compatíveis com a máquina de uso geral	Observação qualitativa e tempos de execução por etapa

Fonte: elaborado pelo autor.

Os critérios CR1 a CR3 correspondem aos aspectos da executabilidade e da integração; os critérios CR4 a CR8 correspondem ao aspecto de adequação aos requisitos de preservação. O critério CR9 é um critério extra de viabilidade, que sustenta a pertinência do workflow ao contexto de instituição pequeno porte.

8.4 LOTES DE OBJETOS DE AVALIAÇÃO

Para exercitar tanto o fluxo nominal quanto os pontos de decisão em falhas, a avaliação foi conduzida sobre objetos digitais organizados em cinco lotes de teste, descritos no Quadro 21. Os lotes combinam objetos digitais obtidos no corpus de objetos digitais do Projeto Format-Corpus da Open Preservation Foundation (Open

Preservation Foundation, 2026) e objetos digitais de controle, estes últimos necessários para reproduzir, de forma segura e replicável, situações de exceção no processamento do acervo para ativação dos três pontos de decisão, como a presença de vírus, arquivos corrompidos ou mal-formados ou falha de replicação. Os lotes L1 e L2 avaliam, respectivamente, a ingestão bem-sucedida e o tratamento de alertas não-fatais; os lotes L3, L4 e L5 isolam, cada um, um dos três pontos de decisão em falhas (Seção 6.4).

A composição dos lotes reflete o perfil de acervo digitalizado típico das instituições de pequeno porte: matrizes de imagem em TIFF, documentos em formatos de suítes de software de escritório e PDF. A escolha de formatos e a estrutura de entrada (diretório bruto de matrizes ou pacote BagIt já constituído) corresponde aos dois formatos de material de entrada previstos na Seção 5.

Quadro 21 — Lotes de objetos para avaliação

Lote	**Finalidade**	**Composição**	**Origem**
L1 — Lote íntegro	Exercitar o fluxo nominal, da etapa E1 a etapa E9	Matriz TIFF bem formada, documentos de suítes de escritório e e-books variados	Format-Corpus da Open Preservation Foundation
L2 — Lote misto	Exercitar alertas sem interrupção (E4 e E5)	O mesmo conjunto do Lote L1, porém acrescido de arquivos não conformes, formatos não reconhecidos e um arquivo TIFF conforme, válido, porém não processável para derivação.	Format-Corpus da Open Preservation Foundation
L3 — Vírus na admissão	Exercitar o primeiro ponto de decisão (E2) e interrupção do workflow	O mesmo conjunto do Lote L1, porém com um arquivo de teste EICAR (vírus de teste padrão).	Format-Corpus da Open Preservation Foundation + EICAR
L4 — Pacote corrompido	Exercitar o segundo ponto de decisão (E3) e interrupção do workflow	Pacote BagIt válido com um byte alterado em objeto do payload após a geração do manifesto.	SIP BagIt válido + alteração controlada
L5 — Destino indisponível	Exercitar o terceiro ponto de decisão (E9)	O mesmo conjunto do Lote L1, com replicação ativada porém com um dos três destinos inválido ou inalcançável.	Format-Corpus da Open Preservation Foundation + destino de replicação inválido

Fonte: elaborado pelo autor (2026).

O arquivo de teste EICAR é uma sequência inócua reconhecida por todos os antivírus como se fosse uma ameaça, adotada como padrão pela indústria justamente para validar a detecção sem expor o sistema a vírus real, sendo a forma segura e reprodutível de exercitar o microsserviço de validação de admissão (MS1) na etapa E2 (EICAR, 2026).

8.5 EXECUÇÃO

Cada lote foi submetido ao workflow por meio da invocação do orquestrador a partir da linha de comando, informando o diretório de entrada, o diretório de saída e, opcionalmente, o nome da sessão e o arquivo de configuração dos destinos de replicação, conforme a convenção descrita na Seção 7.2, um exemplo prático é exibido no Quadro 22:

Quadro 22 — Exemplo de comando para chamada de teste das etapas E1-E9 do workflow

```
python3 orquestrador.py --entrada /entradas/entrada_L1 --saida
/sips/saida --nome teste_L1 --destinos destinos.json
```

Fonte: Elaborado pelo autor (2026)

Em todos os testes foi usado o mesmo arquivo de configuração para os destinos de replicação. O arquivo é um arquivo texto que contém informações sobre os tres destinos de replicação dos AIPs. Nestes testes foram especificados dois diretórios locais e um remoto. O diretório rotulado como “primario_local” é um diretório local no disco da maquina virtual executando os testes. O diretorio rotulado como “secundario_rede_local” é um ponto de montagem de um disco externo, também virtualizado para os testes, e o diretório rotulado como “google drive” é um armazenamento remoto configurado através do rclone em uma conta no google drive, no diretório “workflowtcc/aips”. O conteúdo do arquivo de configuração usado nos testes é mostrado no quadro 23:

Quadro 23 — Arquivo de configuração destinos.json, com destinos de replicação

```
[
  {
    "rotulo": "primario_local",
```

```
    "tipo": "local",
    "caminho": "/compartilhado/aips_primario"
  },
  {
    "rotulo": "secundario_rede_local",
    "tipo": "local",
    "caminho": "/compartilhado/aips_secundario"
  },
  {
    "rotulo": "google_drive",
    "tipo": "remoto",
    "caminho": "gdrive:workflowtcc/aips"
  }
]
```

Fonte: Elaborado pelo autor (2026)

A etapa E10, de verificação periódica de integridade opera fora do fluxo de ingestão e foi acionada separadamente, por execução direta do script verificacao.py sobre os pacotes AIP já armazenados, simulando a verificação periódica do acervo. Um exemplo do comando de chamada da etapa E10 é mostrado no Quando 24:

Quadro 24 — Exemplo de comando para chamada de teste da etapa 10 do workflow

```
python3 verificacao.py --armazenamento /compartilhado/aips_secundario/
```

Fonte: Elaborado pelo autor (2026)

Para cada execução foram coletadas e analisadas as seguintes evidências: a saída de terminal e os registros gravados em logs/; a estrutura de diretórios do pacote gerado; os documentos PREMIS do subdiretório de metadados; os relatórios de identificação e de conformidade; e a saída da validação BagIt. Esses elementos sustentam o juízo sobre cada critério do quadro 20.

8.6 RESULTADOS

Os resultados são apresentados por cenário, do fluxo nominal aos casos de falha, e consolidados no quadro 46. A estrutura de pacote tomada como referência para a inspeção dos produtos é a definida na Seção 7.1, com os subdiretórios data/, metadata/ (contendo premis.xml) e logs/, além dos manifestos e arquivos de controle do BagIt.

8.6.1 Lote L1 — Lote íntegro

O lote L1 envolve exercitar o fluxo nominal, da etapa E1 a etapa E9, com base em matrizes TIFF bem formadas, documentos de suítes de escritório e e-books variados. A execução do workflow sobre o lote L1 percorreu integralmente as etapas E1 a E9. Na recepção (E1), o material foi descompactado a partir de um diretório bruto de 18 matrizes. Em E2, a varredura antivírus (MS1, ClamAV) concluiu sem detecção de ameaças. Em E3, o microsserviço de empacotamento (MS6, BagIt-Python) constituiu o SIP com manifestos SHA-256, estabelecendo a linha de base de fixidez de cada matriz. Em E4, a identificação de formato (MS2, DROID) reconheceu a totalidade dos arquivos pelo registro PRONOM, entre eles a matriz balloon.tif, identificada como Tagged Image File Format (PUID fmt/353). Em E5, a validação de conformidade (MS3, JHOVE) não registrou nenhuma reprovação. Em E6, os metadados técnicos das matrizes foram extraídos (MS4, FITS) e inseridos no PREMIS. Em E7, foram geradas cinco representações derivadas (MS9) — JPEG de qualidade 85 para as imagens e PDF/A para os documentos textuais, sem alteração das matrizes, e os metadados das derivadas foram extraídos. Em E8, o microsserviço de empacotamento consolidou o AIP e validou os manifestos. Em E9, o microsserviço de replicação (MS8) distribuiu o AIP segundo a regra 3-2-1.

A estrutura do AIP reúne, sob data/, as matrizes (originais/) e as derivadas (derivadas/); os metadados de preservação em metadata/ (o premis.xml); e os registros de execução em logs/, incluindo o subdiretório com os logs replicação, além dos manifestos e arquivos de controle do BagIt, conforme apresentado no Quadro 25.

Quadro 25 — Evidência: árvore do AIP (resumida)

```
teste_L1-ddee076ba6dc/
├── bagit.txt
├── bag-info.txt                (Bag-Type: AIP; Payload-Oxum: 34026744.23)
├── manifest-sha256.txt
├── tagmanifest-sha256.txt
├── data/
│   ├── originais/              18 matrizes
│   └── derivadas/               5 derivadas
├── logs/
│   ├── workflow_*.log, admissao_*.log, identificacao_*.csv
│   ├── conformidade_*/*.jhove.xml, caracterizacao_*/, derivadas_*.log
│   └── replicacao_20260602-165417/   primario_local.log,
secundario_rede_local.log, google_drive.log
└── metadata/
    └── premis.xml
```

Fonte: Elaborado pelo autor

Concluída a replicação, o pacote foi revalidado, e a validação BagIt do AIP final confirmou a consistência entre os manifestos SHA-256 e o conteúdo (23 arquivos de payload), demonstrando que a inclusão dos logs e eventos da etapa E9 não comprometeu a integridade do pacote, conforme apresentado no Quadro 26.

Quadro 26 — Evidência: validação BagIt (pós replicação)

```
teste_L1-ddee076ba6dc is valid: True
Payload files: 23 | Algoritmo de manifesto: sha256 | Payload-Oxum:
34026744.23
```

Fonte: Elaborado pelo autor

O documento PREMIS acumulou, em sequência ininterrupta, os onze eventos de preservação correspondentes às etapas executadas, do controle de admissão à replicação, evidenciando a cadeia de custódia exigida como condição de autenticidade (CR6). Cada evento registra seu tipo, data-hora, agente (software e organização) e resultado, e cada representação derivada foi vinculada à sua matriz por relacionamento de derivação, preservando-se a fixidez SHA-256 de cada objeto, conforme apresentado no Quadro 27.

Quadro 27 — Evidência: cadeia de eventos PREMIS, do log do orquestrador

```
evt-0001 | virus check           | success | 16:52:23   (E2)
evt-0002 | creation              | success | 16:52:23   (E3, SIP)
evt-0003 | format identification | success | 16:52:29   (E4)
evt-0004 | format validation     | success | 16:52:34   (E5)
evt-0005 | metadata extraction   | success | 16:53:53   (E6, matrizes)
evt-0006 | migration             | success | 16:53:54   (E7, derivadas)
evt-0007 | metadata extraction   | success | 16:54:17   (E7, derivadas)
evt-0008 | ingestion             | success | 16:54:17   (E8, AIP)
evt-0009 | replication           | success | 16:57:00   (E9, primário
local)
evt-0010 | replication           | success | 16:57:00   (E9, secundário
rede local)
evt-0011 | replication           | success | 16:57:00   (E9, remoto
Google Drive)
```

Fonte: Elaborado pelo autor

Cada representação derivada preserva sua própria fixidez SHA-256 e declara, por relacionamento de derivação (has source), a matriz que lhe deu origem; a ação

de geração foi registrada no evento de migração (evt-0006). O Quadro 28 traz a evidência relativa à fixidez da matriz e a relação matriz-derivada no PREMIS.

Quadro 28 — Evidência: fixidez da matriz e relação matriz-derivada no PREMIS

```
<!-- Matriz: balloon.tif (objeto de origem) -->
<premis:object xsi:type="premis:file">
  <premis:objectIdentifier>

<premis:objectIdentifierValue>data/originais/balloon.tif</premis:objectId
entifierValue>
  </premis:objectIdentifier>
  <premis:objectCharacteristics>
    <premis:format>
      <premis:formatDesignation><premis:formatName>Tagged Image File
Format</premis:formatName></premis:formatDesignation>

<premis:formatRegistry><premis:formatRegistryName>PRONOM</premis:formatRe
gistryName>

<premis:formatRegistryKey>fmt/353</premis:formatRegistryKey></premis:form
atRegistry>
    </premis:format>
    <premis:fixity>
      <premis:messageDigestAlgorithm>SHA-
256</premis:messageDigestAlgorithm>

<premis:messageDigest>2259fe172c3a0fdd41c39d91…</premis:messageDigest>
    </premis:fixity>
  </premis:objectCharacteristics>
</premis:object>

<!-- Derivada: balloon_derivada.jpg (declara a matriz como fonte) -->
<premis:object xsi:type="premis:file">
  <premis:objectIdentifier>

<premis:objectIdentifierValue>data/derivadas/balloon_derivada.jpg</premis
:objectIdentifierValue>
  </premis:objectIdentifier>
  <premis:objectCharacteristics>
    <premis:fixity>
      <premis:messageDigestAlgorithm>SHA-
256</premis:messageDigestAlgorithm>

<premis:messageDigest>1da8fe10426c8a4967703278e56d8211…</premis:messageDi
gest>
    </premis:fixity>
    <premis:size>2530978</premis:size>
  </premis:objectCharacteristics>
  <premis:relationship>
    <premis:relationshipType>derivation</premis:relationshipType>
    <premis:relationshipSubType>has source</premis:relationshipSubType>
    <premis:relatedObjectIdentifier>

<premis:relatedObjectIdentifierType>local</premis:relatedObjectIdentifier
Type>

<premis:relatedObjectIdentifierValue>data/originais/balloon.tif</premis:r
```

```
elatedObjectIdentifierValue>
    </premis:relatedObjectIdentifier>
  </premis:relationship>
</premis:object>
```

Fonte: Elaborado pelo autor

A replicação seguiu a estratégia em duas fases prevista no projeto (Seção 6), as transferências foram executadas sobre o pacote íntegro saído da etapa E8 e, ao final, os logs de cada destino foram gravados em logs/replicacao_20260602-165417/, registrou-se um evento PREMIS replication por destino e o pacote foi revalidado, mantendo o AIP consistente para futuras execuções das etapas E9 e E10. As três cópias foram verificadas após a transferência: As cópias local e de rede local por rsync em modo checksum, e a cópia remota por rclone check, que confirmou ausência de divergências sobre os arquivos do pacote, materializando o requisito de redundância 3-2-1 com integridade verificada em cada destino (CR8), conforme apresentado no Quadro 29.

Quadro 29 — Evidência: eventos de replicação, do log de replicação

```
[success] rsync  :: primario_local        — cópia concluída via rsync -c
(checksum)
[success] rsync  :: secundario_rede_local — cópia concluída via rsync -c
(checksum)
[success] rclone :: google_drive          — transferência remota
verificada por hash (rclone check: 0 differences)
```

Fonte: Elaborado pelo autor

Em síntese, esta execução evidenciou o encadeamento correto das etapas E1 a E9 (CR1, CR2), a validade do pacote AIP após sua construção (CR4), a completude dos metadados técnicos e da cadeia de eventos PREMIS (CR5, CR6), a geração de derivadas vinculadas às matrizes (CR7) e a redundância 3-2-1 com integridade verificada nos três destinos (CR8). O ciclo completo de ingestão concluiu em aproximadamente 4 min 43 s na máquina virtual de uso geral, dos quais cerca de 2 min 43 s correspondem à etapa de replicação E9 (CR9).

8.6.2 Lote L2 — Lote misto

A execução sobre o lote L2, composto por 25 matrizes em formatos variados de documento, imagem e e-book, teve por objetivo verificar o comportamento do workflow diante de arquivos cuja situação é relevante para a preservação, mas que, pela política adotada para instituições de pequeno porte, não deve interromper o fluxo (Seção 6.3.2).

A varredura de admissão (Etapa E2) concluiu sem detecção de ameaças e o pacote percorreu integralmente as etapas até a replicação (Etapa E9). Os três pontos não nominais exercitados foram tratados como alertas documentados, sem suspensão do processamento. O Quadro 30 mostra esses eventos nos registros do log do orquestrador.

Quadro 30 — Evidência: trecho do log do orquestrador (E4, E5 e E7)

```
17:50:44 | WARNING | Arquivos não reconhecidos pelo PRONOM: lorem-
ipsum.rtf, lorem-ipsum.snb
17:50:44 | INFO    | Evento PREMIS registrado: format identification
(success).
17:50:47 | WARNING | Arquivo não conforme: lorem-ipsum.epub — Well-
Formed, but not valid
17:50:50 | INFO    | Validade indeterminada (módulo JHOVE não valida este
formato): lorem-ipsum.fb2 — Well-Formed
17:50:53 | WARNING | Arquivo não conforme: old-style-jpeg-compression.tif
— Well-Formed, but not valid
17:50:54 | INFO    | Evento PREMIS registrado: format validation
(success).
17:52:44 | ERROR   | Comando 'magick' retornou código 1 (incoerência na
tag BitsPerSample): grayscale_8bpp_wrong_bpptag.tif
17:52:46 | INFO    | Evento PREMIS registrado: migration (failure).
```

Fonte: Elaborado pelo autor

Na etapa E4, dois arquivos não foram reconhecidos pelo registro PRONOM: Um arquivo RTF (lorem-ipsum.rtf) e um arquivo SNB (lorem-ipsum.snb). O microsserviço de identificação (MS2, DROID) registrou um alerta e o fluxo prosseguiu, preservando a não identificação como informação relevante: Na saída CSV do DROID esses arquivos constam com PUID e nome de formato vazios, e no PREMIS o objeto correspondente recebe formatName igual a unknown. Essa documentação da não identificação como metadado de preservação do próprio objeto é evidenciada no Quadro 31.

Quadro 31 — Evidência: objeto PREMIS de arquivo não identificado (trecho):

```
<premis:object xsi:type="premis:file">
  <premis:objectIdentifier>
    <premis:objectIdentifierValue>data/originais/lorem-
ipsum.rtf</premis:objectIdentifierValue>
  </premis:objectIdentifier>
  <premis:objectCharacteristics>
    <premis:size>6891</premis:size>
    <premis:format>

<premis:formatDesignation><premis:formatName>unknown</premis:formatName><
/premis:formatDesignation>
    </premis:format>
  </premis:objectCharacteristics>
</premis:object>
```

Fonte: Elaborado pelo autor

Na etapa E5, a validação de conformidade (MS3, JHOVE) atestou dois arquivos como "bem formados, porém não válidos" (Well-Formed, but not valid): Um arquivo EPUB (lorem-ipsum.epub), cujo erro RSC-005 aponta um identificador inválido em usa estrutura, e uma imagem TIFF (old-style-jpeg-compression.tif), reprovada pela mensagem TIFF-HUL-49 por compressão JPEG sem o parâmetro JPEGProc.

Outros arquivos foram ainda classificados como de validade não determinada por ausência de módulo validador específico no JHOVE (por exemplo, o arquivo FB2). Em todos os casos foram gerados alertas e o evento de validação foi concluído com resultado success, sem suspensão do fluxo. O parecer técnico do JHOVE para o TIFF reprovado, preservado no AIP, evidencia o nível de detalhe documentado, conforme o Quadro 32.

Quadro 32 — Evidência: parecer JHOVE do modulo TIFF-hul de não conformidade (trecho)

```
<reportingModule release="1.9.5" date="2024-08-22">TIFF-
hul</reportingModule>
<format>TIFF</format>
<version>6.0</version>
<status>Well-Formed, but not valid</status>
<messages>
  <message severity="error" id="TIFF-HUL-49">JPEGProc not defined for
JPEG compression</message>
</messages>
```

Fonte: Elaborado pelo autor

Na etapa E7, um terceiro tipo de ocorrência não-fatal foi exercitado: a geração da derivada da imagem grayscale_8bpp_wrong_bpptag.tif falhou. Cabe notar que

esse arquivo havia sido atestado como conforme pelo JHOVE em E5 (Well-Formed and valid), e a falha ocorreu apenas na etapa E7, quando a ferramenta ImageMagick tentou decodificar os pixels para gerar a derivada e encontrou uma incoerência entre características declaradas e os dados de imagem efetivos. Esse evento ilustra uma distinção relevante em preservação digital: A conformidade estrutural ao formato, verificada pela validação, não garante que o objeto seja processável com sucesso por toda e qualquer ferramenta, e justifica que o workflow trate a falha de derivação como um alerta próprio, independente do veredito de conformidade da etapa E5. A falha foi documentada em um evento PREMIS de migração com resultado failure, e o microsserviço prosseguiu com os demais arquivos: Das 25 matrizes, nove geraram derivadas, quinze foram ignoradas por não comportarem derivada de acesso e apenas uma falhou. O registro desse evento no PREMIS, ainda que um resultado do tipo failure, integra-se à cadeia de eventos sem interromper o fluxo, e pode ser observado no Quadro 33.

Quadro 33 — Evidência: evento PREMIS de migração com resultado failure (trecho)

```
<premis:event>
  <premis:eventIdentifier>
    <premis:eventIdentifierValue>teste_L2-42e21385e9ef-evt-
0006</premis:eventIdentifierValue>
  </premis:eventIdentifier>
  <premis:eventType>migration</premis:eventType>
  <premis:eventDateTime>2026-06-02T17:52:46-03:00</premis:eventDateTime>
  <premis:eventOutcomeInformation>
    <premis:eventOutcome>failure</premis:eventOutcome>
    <premis:eventOutcomeDetail>
      <premis:eventOutcomeDetailNote>Derivadas geradas: 9; ignoradas
(tipo não aplicável): 15; falhas: 1.</premis:eventOutcomeDetailNote>
    </premis:eventOutcomeDetail>
  </premis:eventOutcomeInformation>
  <premis:linkingAgentIdentifier>

<premis:linkingAgentIdentifierValue>ImageMagick/LibreOffice/FFmpeg</premi
s:linkingAgentIdentifierValue>
    <premis:linkingAgentRole>software</premis:linkingAgentRole>
  </premis:linkingAgentIdentifier>
</premis:event>
```

Fonte: Elaborado pelo autor

O fluxo seguiu até o empacotamento do AIP (E8) e a replicação 3-2-1 (E9), comprovando que uma falha pontual de transformação não compromete a preservação das matrizes nem a conclusão da ingestão. Os três casos demonstram,

em conjunto, o atendimento ao critério CR3 na vertente de alertas e confirmam a decisão de projeto de documentar falhas de identificação, não conformidades e falhas pontuais de derivação sem decidir automaticamente sobre rejeição ou migração, conduta que pressuporia uma política formal de preservação ausente no perfil institucional de pequeno porte. Como no lote L1, em todas as ocorrências as informações foram preservadas como metadados de preservação, assegurando rastreabilidade (CR6) sem interrupção do fluxo. As notas de resultado dos três eventos evidenciam o que foi documentado, conforme pode ser visto no Quadro 34.

Quadro 34 — Evidência: notas de resultado dos eventos de E4, E5 e E7 (PREMIS)

```
[success] format identification — DROID PRONOM
  "Identificação concluída com 2 arquivo(s) não reconhecido(s) (alerta):
lorem-ipsum.rtf, lorem-ipsum.snb."

[success] format validation — JHOVE
  "Validação concluída (25 arquivo(s)); 2 não conforme(s) documentado(s):
lorem-ipsum.epub, old-style-jpeg-compression.tif."

[failure] migration — ImageMagick/LibreOffice/FFmpeg
  "Derivadas geradas: 9; ignoradas (tipo não aplicável): 15; falhas: 1."
```

Fonte: Elaborado pelo autor

Estes três casos demonstram o atendimento ao critério CR3 na vertente de alertas e confirmam a decisão de projeto de documentar falhas de identificação, não conformidades e falhas pontuais de derivação sem decidir automaticamente sobre rejeição ou migração, uma conduta que pressuporia uma política formal de preservação ausente no perfil institucional alvo. Em todos os casos, essas informações foram preservadas como metadados de preservação (em eventos PREMIS), assegurando rastreabilidade (CR6) sem interrupção do fluxo.

8.6.3 Lote L3 — Vírus na admissão (ponto de decisão E2)

O lote L3 avalia o primeiro ponto de decisão, na etapa E2, de validação de admissão. Submeteu-se um lote contendo o arquivo de teste EICAR (eicar.com), reconhecido pelos antivírus como uma assinatura de ameaça sem constituir código malicioso real (Seção 8.4). Na etapa E2, o microsserviço de validação de admissão

(MS1, ClamAV) varreu a cópia de trabalho e detectou a assinatura, retornando código de erro, conforme apresentado no Quadro 35.

Quadro 35 — Evidência: log de saída do ClamAV (E2)

```
clamscan -r --infected --no-summary .../_trabalho/teste_L3-eb24348e6903
.../teste_L3-eb24348e6903/eicar.com: Eicar-Test-Signature FOUND
--- exit code: 1 ---
```

Fonte: Elaborado pelo autor

Pela ocorrência da detecção de vírus, o workflow executou o comportamento previsto para o primeiro ponto de decisão: moveu a totalidade do material recebido para um diretório de quarentena como medida de contenção, registrou um evento PREMIS de rejeição e interrompeu o fluxo, sem iniciar a etapa E3 que constituiria o SIP intermediário. O log da sessão documenta a sequência da decisão, como pode ser visto no Quadro 36.

Quadro 36 — Evidência: trecho do log do orquestrador (E2)

```
19:49:10 | WARNING | Arquivos infectados detectados. Material rejeitado.
19:49:10 | WARNING | Material movido para quarentena:
.../_trabalho/quarentena/teste_L3-eb24348e6903
19:49:10 | INFO    | Evento PREMIS registrado: virus check (failure).
19:49:10 | ERROR   | FLUXO INTERROMPIDO em E2: material rejeitado na
admissão.
19:49:10 | INFO    | Encerrando sessão de ingestão.
```

Fonte: Elaborado pelo autor

Embora nenhum pacote AIP tenha sido gerado, o workflow produziu, ainda assim, um registro PREMIS documentando a rejeição na área de saída da sessão. Esse comportamento é relevante para a rastreabilidade (CR6): A recusa de um objeto não é silenciosa, mas auditável, registrando o tipo de evento, o agente que o executou, a data-hora e o motivo da rejeição, com o identificador do item afetado, conforme mostrado no Quadro 37.

Quadro 37 — Evidência: evento PREMIS de rejeição (trecho)

```
<premis:event>
  <premis:eventIdentifier>
    <premis:eventIdentifierValue>teste_L3-eb24348e6903-evt-
```

```
0001</premis:eventIdentifierValue>
  </premis:eventIdentifier>
  <premis:eventType>virus check</premis:eventType>
  <premis:eventDateTime>2026-06-02T19:49:10-03:00</premis:eventDateTime>
  <premis:eventOutcomeInformation>
    <premis:eventOutcome>failure</premis:eventOutcome>
    <premis:eventOutcomeDetail>
      <premis:eventOutcomeDetailNote>Material rejeitado por contaminação.
Itens:
eicar.com: Eicar-Test-Signature FOUND</premis:eventOutcomeDetailNote>
    </premis:eventOutcomeDetail>
  </premis:eventOutcomeInformation>
  <premis:linkingAgentIdentifier>
    <premis:linkingAgentIdentifierValue>ClamAV
1.4.4</premis:linkingAgentIdentifierValue>
    <premis:linkingAgentRole>software</premis:linkingAgentRole>
  </premis:linkingAgentIdentifier>
</premis:event>
```

Fonte: Elaborado pelo autor

O fato de o workflow ter sido interrompido antes de alcançar as etapas E3 a E9 é, neste cenário, o resultado esperado: o workflow não processa e também não armazena objeto cujo estado de segurança é incerto, em conformidade com a estratégia de falha explícita (Seção 6.4). A contenção do material em quarentena, em vez de seu descarte automático, preserva a possibilidade de triagem humana posterior, coerente com a política de não automatizar decisões que excedam a contenção no perfil de instituição de pequeno porte.

8.6.4 Lote L4 — Pacote corrompido (ponto de decisão E3)

O lote L4 avalia o segundo ponto de decisão, na validação de um pacote SIP BagIt recebido como objeto de entrada no workflow. Um pacote SIP BagIt válido com os mesmos arquivos do Lote íntegro L1 teve um de seus arquivos corrompido através da alteração de um byte no conteúdo de um de seus arquivos. Por se tratar de material livre de contaminação por vírus, a varredura de admissão (Etapa E2) foi aprovada, permitindo que o fluxo alcançasse o segundo ponto de decisão. Na etapa E3, ao validar o pacote BagIt recebido, o microsserviço de empacotamento (MS6) detectou que o arquivo data/originais/lorem-ipsum.pdf não correspondia ao checksum registrado, um claro sinal de que o pacote foi corrompido após sua constituição original. Diante da divergência, foi registrado um evento PREMIS de verificação de fixidez com resultado failure e o fluxo foi interrompido, sem constituir o AIP, mantendo

o material original no diretório de entrada para reprocessamento, o que pode ser visto no Quadro 38.

Quadro 38 — Evidência: trecho do log do orquestrador (E2 aprovada, E3 reprovada)

```
19:19:09 | INFO  | Varredura concluída: nenhuma ameaça detectada.
19:19:09 | INFO  | Evento PREMIS registrado: virus check (success).
19:19:09 | INFO  | === E3 — Constituição ou validação do SIP (MS6) ===
19:19:09 | DEBUG | PREMIS existente carregado (11 eventos).
19:19:09 | ERROR | Validação do SIP falhou: Bag validation failed:
data/originais/lorem-ipsum.pdf sha256 validation failed:
expected="b55fd1597a4f1a91ea0c02e8571610541ccaf1aa02b68000726b419afe407ea
8"
found="a0c16d5806a1070d5519679edaaebe4420bb1216214dcbfed957685a3337544b"
19:19:09 | INFO  | Evento PREMIS registrado: fixity check (failure).
19:19:09 | ERROR | FLUXO INTERROMPIDO em E3. O material original
permanece na entrada.
```

Fonte: Elaborado pelo autor

Como o material chegou na forma de um pacote SIP já constituído, sua cadeia de eventos de preservação anterior e todos os eventos PREMIS — onze neste caso — registrados em sua constituição e processamento original foram reconhecidos pelo workflow, e o evento PREMIS de failure em validação de fixidez foi agregado ao conjunto, mantendo a continuidade dessa cadeia de eventos, preservando a rastreabilidade do que ocorreu com o objeto (CR6), conforme mostrado no Quadro 39.

Quadro 39 — Evidência: evento PREMIS de falha em verificação de fixidez (trecho)

```
<premis:event>
  <premis:eventIdentifier>    <premis:eventIdentifierValue>teste_L4-
b454c0a18a09-evt-0012</premis:eventIdentifierValue>
  </premis:eventIdentifier>
  <premis:eventType>fixity check</premis:eventType>
  <premis:eventDateTime>2026-06-04T19:19:09-03:00</premis:eventDateTime>
  <premis:eventOutcomeInformation>
    <premis:eventOutcome>failure</premis:eventOutcome>
    <premis:eventOutcomeDetail>
      <premis:eventOutcomeDetailNote>Falha de validação do pacote BagIt
recebido: Bag validation failed: data/originais/lorem-ipsum.pdf sha256
validation failed: expected="b55fd159…ea8"
found="a0c16d58…44b"</premis:eventOutcomeDetailNote>
    </premis:eventOutcomeDetail>
  </premis:eventOutcomeInformation>
  <premis:linkingAgentIdentifier>
    <premis:linkingAgentIdentifierValue>BagIt-Python
1.9.0</premis:linkingAgentIdentifierValue>
    <premis:linkingAgentRole>software</premis:linkingAgentRole>
  </premis:linkingAgentIdentifier>
</premis:event>
```

Fonte: Elaborado pelo autor

A detecção da adulteração na admissão evidencia o papel do empacotamento em BagIt como mecanismo de garantia de integridade: O pacote carrega em seus manifestos SHA-256 o checksum esperado de cada arquivo, de modo que qualquer divergência entre o declarado nos manifestos e o efetivamente analisado no conteúdo é identificável imediatamente na etapa E3, com a precisão de indicar o arquivo afetado. Tal como no primeiro ponto de decisão, o fluxo não prossegue com objeto cujo estado de integridade é incerto, em conformidade com a estratégia de falha explícita (Seção 6.4). Diferente do caso do primeiro ponto, porém, o material não é movido para quarentena, mas mantido integralmente no diretório de entrada para que seus logs e metadados PREMIS possam ser investigados para correção de problemas antes de nova tentativa de ingestão.

8.6.5 Lote L5 — Destino indisponível (ponto de decisão E9)

O lote L5 avalia o terceiro ponto de decisão, na etapa de armazenamento e replicação. A entrada recebe um lote íntegro e válido — aprovado em E2 e E3 — com a replicação ativada para três destinos (cópia primária local, cópia secundária em rede local e cópia remota em nuvem), tornando-se deliberadamente inacessível o destino secundário, simulado por um caminho de rede inacessível. Diferentemente dos pontos de decisão anteriores, o terceiro não interrompe o fluxo. Na etapa E9, o microsserviço de replicação (MS8) transferiu o AIP para os dois destinos disponíveis e, ao tentar o destino indisponível, registrou a falha relativa àquele destino sem suspender a operação. Os três eventos PREMIS de replicação documentam o resultado para cada destino, o que pode ser observado no Quadro 40.

Quadro 40 — Evidência: resultado por destino (eventos no log de replicação)

```
success] rsync  :: primario_local        — cópia concluída via rsync -c
(checksum)
[failure] rsync  :: secundario_rede_local — sem permissão para criar
destino (Permission denied)
[success] rclone :: google_drive          — transferência remota
verificada (rclone check: 0 diferenças, 55 arquivos)
```

Fonte: Elaborado pelo autor

Concluída a etapa, o workflow emitiu um alerta relativo ao destino e prosseguiu até o encerramento normal da sessão, com o AIP devidamente constituído, que pode ser visto no Quadro 41.

Quadro 41 — Evidência: trecho do log do orquestrador (E9)

```
21:14:51 | INFO    | Evento PREMIS registrado: replication (success).
21:14:51 | INFO    | Evento PREMIS registrado: replication (failure).
21:14:51 | INFO    | Evento PREMIS registrado: replication (success).
21:14:51 | WARNING | Replicação concluída com falhas em:
secundario_rede_local
21:14:51 | WARNING | E9 concluída com falha em ao menos um destino
(alerta relativo ao destino).
21:14:51 | INFO    | Fluxo de ingestão concluído. AIP em: .../teste_L5-
1ac49efda2d8
```

Fonte: Elaborado pelo autor

O evento de falha registra, de forma autônoma e auditável, o destino afetado e a causa precisa da indisponibilidade, sem comprometer os eventos de sucesso dos demais destinos (CR6), o que pode ser observado no Quadro 42.

Quadro 42 — Evidência: evento PREMIS replication com resultado failure (trecho)

```
<premis:event>
  <premis:eventIdentifier>
    <premis:eventIdentifierValue>teste_L5-1ac49efda2d8-evt-
0010</premis:eventIdentifierValue>
  </premis:eventIdentifier>
  <premis:eventType>replication</premis:eventType>
  <premis:eventDateTime>2026-06-02T21:14:51-03:00</premis:eventDateTime>
  <premis:eventOutcomeInformation>
    <premis:eventOutcome>failure</premis:eventOutcome>
    <premis:eventOutcomeDetail><premis:eventOutcomeDetailNote>
      Destino 'secundario_rede_local' (local,
/mnt/servidor_remoto/aips_secundario):
      Sem permissão para criar destino [Errno 13] Permission denied.
    </premis:eventOutcomeDetailNote></premis:eventOutcomeDetail>
  </premis:eventOutcomeInformation>
  <premis:linkingAgentIdentifier>

<premis:linkingAgentIdentifierValue>rsync</premis:linkingAgentIdentifierV
alue>
    <premis:linkingAgentRole>software</premis:linkingAgentRole>
  </premis:linkingAgentIdentifier>
</premis:event>
```

Fonte: Elaborado pelo autor

Por fim, a validação do AIP confirmou que o objeto preservado permaneceu íntegro (23 arquivos de *payload*, manifestos consistentes), apesar da falha em um dos destinos. A redundância foi alcançada nos dois destinos disponíveis e o objeto não foi comprometido. Este cenário demonstra a degradação graciosa prevista para o terceiro ponto de decisão. Ao contrário dos pontos 1 e 2, onde a segurança ou a integridade do próprio objeto está em dúvida e o fluxo é interrompido, aqui o pacote AIP preservado está íntegro, e a indisponibilidade de um destino é tratada como alerta a ser sanado em nova tentativa de replicação, sem recusar o objeto nem desfazer as cópias bem-sucedidas (CR3, CR8). A reconciliação posterior da cópia faltante pode ser conduzida por nova execução da replicação ou detectada pela verificação periódica de integridade (E10).

8.6.6 Verificação periódica de integridade (Etapa E10)

A etapa E10 opera fora do fluxo de ingestão, sobre os pacotes AIP já armazenados, e materializa a manutenção contínua que distingue a preservação digital do mero armazenamento. Para a avaliação, o microsserviço de verificação (MS5) foi executado sobre um repositório de destino contendo cinco AIP previamente ingeridos, em dois momentos: primeiro com todos os pacotes íntegros e, em seguida, após a alteração deliberada de um arquivo em um dos pacotes, de modo a exercitar tanto a confirmação de integridade quanto a detecção de perda.

Na primeira execução, a verificação calculou os checksums do conteúdo de cada AIP e confirmou a correspondência com os respectivos manifestos SHA-256, registrando um evento PREMIS de verificação bem-sucedida em cada pacote e concluindo sem divergências, o que pode ser observado no Quadro 43.

Quadro 43 — Evidência: log do orquestrador da verificação (primeira execução)

```
22:12:41 | INFO | Iniciando verificação periódica de 5 AIP(s).
22:12:41 | INFO | Evento PREMIS registrado: fixity check (success).
22:12:41 | INFO | AIP 'teste_L1-ae2f848d791c': Íntegro: checksums do
payload conferem com o manifesto SHA-256.
   [... demais AIPs íntegros ...]
22:12:41 | INFO | Verificação concluída: todos os AIPs estão íntegros.
```

Fonte: Elaborado pelo autor

Em seguida foi feita uma alteração diretamente em um byte do conteúdo do arquivo data/originais/lorem-ipsum.pdf no pacote teste_L1-dad219958f68, simulando uma corrupção silenciosa, como a que pode ocorrer como efeito de degradação de mídia ou de adulteração, e a verificação foi executada novamente. Nesta segunda execução foram novamente calculados e comparados os checksums calculados com os checksums dos manifestos, e identificada uma divergência, apontando precisamente o arquivo afetado. Como na validação de admissão do lote L4, a verificação periódica confronta o checksum SHA-256 de cada arquivo com o manifesto, localizando o objeto corrompido dentro do pacote; a diferença é o momento e o alvo: enquanto na etapa E3 a checagem é feita na recepção de pacote, na etapa periódica E10 ela recai periodicamente sobre os pacotes AIP já armazenados, detectando perdas que surjam entre uma verificação e a próxima. O relatório de integridade consolidado registrou quatro pacotes íntegros e um divergente, conforme observado no Quadro 44.

Quadro 44 — Evidência: relatório CSV de integridade (segunda execução)

```
aip;status;mensagem
teste_L1-ae2f848d791c;integro;Íntegro: checksums do payload conferem com
o manifesto SHA-256.
teste_L1-c46ae16a0a57;integro;Íntegro: checksums do payload conferem com
o manifesto SHA-256.
teste_L1-dad219958f68;divergente;DIVERGÊNCIA DE INTEGRIDADE (payload):
checksum divergente: data/originais/lorem-ipsum.pdf
teste_L1-ddee076ba6dc;integro;Íntegro: checksums do payload conferem com
o manifesto SHA-256.
teste_L2-42e21385e9ef;integro;Íntegro: checksums do payload conferem com
o manifesto SHA-256.
```

Fonte: Elaborado pelo autor

A divergência foi registrada como um evento PREMIS de verificação de fixidez com resultado failure, acrescentado à cadeia de eventos do próprio pacote como continuidade de sua história de preservação, de modo que o documento PREMIS passa a registrar tanto as verificações bem-sucedidas quanto a detecção da perda, com data-hora e identificação do arquivo afetado (CR6), conforme observado no Quadro 45.

Quadro 45 — Evidência: evento PREMIS de falha de verificação de fixidez (trecho)

```
<premis:event>
  <premis:eventIdentifier>
    <premis:eventIdentifierValue>teste_L1-dad219958f68-evt-
0013</premis:eventIdentifierValue>
  </premis:eventIdentifier>
  <premis:eventType>fixity check</premis:eventType>
  <premis:eventDateTime>2026-06-02T22:22:33-03:00</premis:eventDateTime>
  <premis:eventOutcomeInformation>
    <premis:eventOutcome>failure</premis:eventOutcome>
    <premis:eventOutcomeDetail>
      <premis:eventOutcomeDetailNote>DIVERGÊNCIA DE INTEGRIDADE
(payload): checksum divergente: data/originais/lorem-
ipsum.pdf</premis:eventOutcomeDetailNote>
    </premis:eventOutcomeDetail>
  </premis:eventOutcomeInformation>
  <premis:linkingAgentIdentifier>
    <premis:linkingAgentIdentifierValue>BagIt-
Python</premis:linkingAgentIdentifierValue>
    <premis:linkingAgentRole>software</premis:linkingAgentRole>
  </premis:linkingAgentIdentifier>
</premis:event>
```

Fonte: Elaborado pelo autor

Diante da divergência, a verificação sinalizou o estado de não conformidade do pacote em dois registros independentes: o relatório de integridade consolidado, que o marcou como divergente, e o evento PREMIS de verificação de fixidez com resultado failure, acrescentado à cadeia do próprio pacote. A redundância estabelecida na etapa E9 fornece as cópias íntegras a partir das quais o objeto corrompido pode ser restaurado, e a verificação periódica é o mecanismo que gera relatórios e eventos que podem alertar sobre a necessidade dessa recuperação. Cada execução acrescenta ao PREMIS de cada AIP um registro datado da verificação, de modo que o histórico de verificações de integridade integra a própria proveniência do objeto. O conjunto evidencia o atendimento aos requisitos de integridade e de detecção de perdas dos NDSA Levels (níveis 1 e 2) na operação continuada do acervo (CR6), demonstrando que o workflow não se limita à ingestão, mas sustenta a vigilância da integridade ao longo do tempo.

8.6.7 Síntese dos resultados

Os resultados relatados por cenário nas subseções anteriores são consolidados no Quadro 46, que cruza os nove critérios de avaliação estabelecidos na seção 8.3 (CR1 a CR9) com os cinco lotes de objetos de avaliação (L1 a L5) e com

a verificação periódica de integridade (E10). A leitura por linha evidencia o comportamento de cada critério nos diferentes cenários; a leitura por coluna, o subconjunto de critérios aplicável a cada cenário. Como cada um percorre o fluxo até um ponto distinto e ativa um conjunto próprio de etapas e de pontos de decisão, nem todos os critérios se aplicam a todos os cenários e as células correspondentes a estas situações recebem um traço (—) previsto na legenda, indicando não aplicabilidade, e não falha de atendimento.

Quadro 46 — Síntese do atendimento aos critérios de avaliação

Critério	**L1**	**L2**	**L3 (E2)**	**L4 (E3)**	**L5 (E9)**	**E10**
CR1 Executabilidade	✓	✓	✓	✓	✓	✓
CR2 Integração	✓	✓	—	—	✓	—
CR3 Pontos de decisão	—	✓	✓	✓	✓	—
CR4 Validade do pacote	✓	✓	—	—	✓	—
CR5 Completude metadados	✓	✓	—	—	✓	—
CR6 Rastreabilidade	✓	✓	✓	✓	✓	✓
CR7 Derivadas	✓	✓	—	—	✓	—
CR8 Redundância	✓	—	—	—	✓*	✓
CR9 Viabilidade hardware	✓	✓	✓	✓	✓	✓

Fonte: elaborado pelo autor
Legenda: ✓ atendido; — não aplicável ao cenário; * redundância parcial (2 de 3 destinos), com a indisponibilidade documentada.

8.7 ANÁLISE DOS RESULTADOS

A análise dos resultados é conduzida à luz dos três aspectos de verificação e da pergunta de pesquisa.

Nos aspectos da executabilidade e de integração, os resultados indicam que o conjunto de ferramentas de software livre selecionado na seção 5 e instrumentalizado pelos scripts da seção 7 executam em ambiente de uso geral, sem a necessidade de um servidor dedicado.

Nesse ambiente de uso geral o orquestrador encadeia os microsserviços na sequência projetada, com os três pontos de decisão operando como especificado: Interrompendo o fluxo na detecção de uma ameaça (L3) ou na detecção de adulteração do pacote (L4) e degradando graciosamente diante da indisponibilidade de destinos de replicação (L5). Esse resultado sustenta a tese de que é possível instrumentalizar microsserviços essenciais de preservação digital com software livre executável em equipamentos comuns, contornando as barreiras de custo e infraestrutura que ferramentas integradas de mercado impõem às instituições de pequeno porte (Seção 1).

No aspecto da adequação aos requisitos de preservação, os objetos e pacotes produzidos pelo workflow evidenciam aderência aos requisitos formalizados pelos modelos conceituais. A constituição de SIP e AIP válidos em BagIt e a sua replicação atendem ao modelo funcional do OAIS quanto aos pacotes de informação e ao armazenamento. A cadeia ininterrupta de eventos PREMIS — incluindo o registro auditável de rejeições e falhas — materializa o requisito de cadeia de custódia documentada que o InterPARES vincula à autenticidade. A verificação de integridade por checksums, a presença de múltiplas cópias e o registro sistemático de metadados situam o resultado do workflow nos níveis 1 e 2 dos NDSA Levels nas categorias de armazenamento, integridade, metadados e conteúdo.

No aspecto da viabilidade, os resultados apontam que a execução em hardware comum ou servidores simples de baixo custo é compatível com a operação por um profissional não especializado em tecnologia da informação, especialmente pela automação por linha de comando, com a ressalva de que decisões que exigem decisões formais de preservação — periodicidade da verificação e topologia de replicação — permanecem, por concepção, a cargo da instituição, e podem ser implementadas através do serviço de agendamento de tarefas Linux e por um arquivo de configuração JSON.

Em conjunto, os resultados indicam que o workflow proposto é exequível e produz objetos digitais preservados com integridade, autenticidade e rastreabilidade verificáveis, respondendo afirmativamente à pergunta de pesquisa para o perfil da instituição de pequeno porte.

9 CONCLUSÃO

Este trabalho partiu da observação de um distanciamento entre a literatura especializada em preservação digital e a realidade de uma parcela significativa das instituições brasileiras de memória. Enquanto boa parte dos modelos conceituais de referência e as ferramentas de mercado, proprietárias ou de código aberto, pressupõem equipes especializadas, infraestrutura dedicada e maturidade organizacional elevada, há um amplo conjunto de instituições arquivísticas, arquivos comunitários, centros documentais universitários, acervos de associações civis e arquivos pessoais que custodiam patrimônio documental relevante operando com equipes não especializadas, orçamentos reduzidos e equipamentos de TI de uso geral. Diante desse quadro, o autor formulou a pergunta que orientou todo o trabalho: Como estruturar um ambiente computacional, composto por microsserviços essenciais de preservação digital, apoiado em software livre, que seja exequível para instituições de memória de pequeno porte?

O percurso das seções anteriores e os resultados da avaliação permitem responder a essa pergunta de forma afirmativa.

O objetivo geral de propor e elaborar esse ambiente computacional e fluxo de trabalho concretizou-se por meio do encadeamento dos cinco objetivos específicos, cujo percurso detalhado consta da metodologia (Seção 3.2). Em síntese, a análise de oito modelos conceituais identificou nove microsserviços essenciais, além de dois contextuais, registrados para evolução futura (Seção 4). Mais do que enumerar serviços, esse resultado evidencia que, em meio à diversidade de microsserviços descritos na literatura e incorporados às ferramentas existentes, é possível delimitar um núcleo essencial e estável: os critérios de seleção entre os oito modelos permitiu separar o que é estrutural do que é acessório, conferindo concisão e objetividade ao emprego de microsserviços em instituições de memória de pequeno porte.

Para cada microsserviço essencial selecionou-se uma ou mais ferramentas de software livre da plataforma Linux, segundo seis critérios derivados dos cinco aspectos expostos na caracterização da instituição de pequeno porte (Seção 5). Para além da seleção, esse resultado demonstra que o núcleo essencial já dispõe de implementação: a função de cada microsserviço, isoladamente, é realizável por bibliotecas e aplicativos livres simples, presentes ou facilmente instaláveis em um ambiente Linux comum, sem que a instituição precise desenvolver suas próprias

ferramentas de preservação nem adquirir soluções integradas complexas. O esforço, assim, desloca-se da construção de ferramentas para a sua combinação criteriosa.

Esses microsserviços e ferramentas foram organizados em um workflow de dez etapas, com três pontos de decisão e estratégia de falha explícita (Seção 6). A lógica do fluxo foi implementada através de scripts Python, com um script orquestrador e scripts para cada microsserviço (Seção 7). O conjunto foi submetido a uma avaliação funcional com cinco lotes de objetos digitais (Seção 8). O alcance dessa etapa não se esgota no encadeamento: a contribuição está em demonstrar como ferramentas heterogêneas, concebidas de forma isolada, podem ser integradas em um fluxo coeso. Essa integração foi obtida por uma camada de scripts em Python que encapsula cada implementação, padroniza suas entradas e saídas e a expõe a um orquestrador, de modo que microsserviços de origens distintas passam a operar como um único workflow, com pontos de decisão e tratamento de falhas explícitos.

Os resultados da avaliação (Seção 8) respondem afirmativamente à pergunta de pesquisa. Demonstrou-se que é possível instrumentalizar os microsserviços essenciais de preservação digital com software livre executável em equipamentos comuns, de uso geral, contornando as barreiras de custo e infraestrutura que as ferramentas integradas de mercado impõem. O produto final gerado pelo workflow — um pacote AIP válido em formato BagIt, com metadados técnicos, uma cadeia ininterrupta de eventos PREMIS em formato XML PREMIS válido, representações derivadas vinculadas às suas matrizes e cópias replicadas com integridade verificada — evidenciou a aderência aos requisitos formalizados pelos modelos conceituais: o modelo funcional e a arquitetura de pacotes do OAIS, o requisito de cadeia de custódia documentada que o InterPARES vincula à autenticidade, e o patamar mínimo de preservação responsável definido pelos níveis 1 e 2 dos NDSA Levels.

A contribuição científica do trabalho está em endereçar uma lacuna da literatura, que documenta predominantemente cenários de maturidade tecnológica elevada e oferece pouca orientação para contextos de escassez de recursos, demonstrando a viabilidade de uma abordagem de parcimônia digital: As funções críticas de preservação podem ser extraídas dos modelos e das ferramentas mais complexas e reorganizadas em um conjunto mínimo, modular e de baixo custo, sem perda de aderência aos requisitos essenciais de integridade, autenticidade e rastreabilidade.

A contribuição prática do trabalho está em oferecer um caminho de entrada acessível para um segmento institucional que custodia parte expressiva do patrimônio documental brasileiro e que, de outro modo, permaneceria à margem de práticas sistemáticas de preservação digital. O workflow proposto não se apresenta como substituto dos modelos conceituais de referência nem das ferramentas integradas de gestão de acervos digitais, mas como um ponto de partida que pode ser progressivamente estendido à medida que a maturidade institucional avança, pois foi implementado integralmente em software livre, com código aberto.

Os resultados devem ser interpretados à luz das limitações da pesquisa. A avaliação teve natureza funcional, focada em demonstrar a executabilidade do workflow e a adequação de seus produtos aos requisitos de preservação, e não constituiu uma validação em ambiente institucional real, com equipe e acervo reais, ao longo de um período prolongado de operação.

O ambiente pressupõe, ainda, uma capacidade técnica mínima para a instalação das ferramentas e mínima familiaridade com a operação por linha de comando, e deixa deliberadamente fora de seu escopo funções como o monitoramento de obsolescência de formatos e a geração de pacotes de disseminação, além de adotar um nível básico de descrição arquivística. Por fim, as decisões que dependem de política formal de preservação, como periodicidade das verificações de integridade, topologia de replicação e conduta diante de arquivos não conformes, permanecem, por concepção, a cargo da instituição.

Essas limitações apontam caminhos para trabalhos futuros. O mais relevante seria a validação do workflow em uma instituição de memória de pequeno porte real, acompanhando sua operação por profissionais não especializados em tecnologia da informação e sobre um acervo efetivo, de modo a avaliar sua usabilidade e sustentabilidade ao longo do tempo. Também importante seria o desenvolvimento de uma interface gráfica simplificada, que poderia reduzir ainda mais a barreira técnica de adoção.

A implementação dos microsserviços classificados como contextuais ampliaria a cobertura funcional do fluxo conforme a maturidade institucional crescesse, assim como a evolução progressiva do empacotamento em direção a especificações mais elaboradas, como o CSIP baseado em METS. Seria igualmente útil o empacotamento da própria solução em um formato de distribuição e instalação simplificadas, reduzindo o esforço de configuração inicial.

Conclui-se que instituições de memória de pequeno porte, frequentemente percebidas como incapazes de praticar preservação digital por carência de recursos, podem dar passos consistentes nessa direção com o apoio de software livre e de um fluxo de trabalho parcimonioso. Ao reduzir a desproporção entre o volume de acervo custodiado e a capacidade institucional de preservá-lo, o workflow proposto contribui para que parte importante do patrimônio documental brasileiro hoje vulnerável possa ser preservada com integridade, autenticidade e rastreabilidade verificáveis, ainda que em seu patamar mínimo responsável.

REFERÊNCIAS

ABRAMS, Stephen; KUNZE, John; LOY, David. An emergent micro-services approach to digital curation infrastructure. **International Journal of Digital Curation**, Edinburgh, v. 5, n. 1, p. 172-186, 2010. Disponível em: https://ijdc.net/index.php/ijdc/article/view/151. Acesso em: 7 abr. 2026.

ABRAMS, Stephen; MORRISSEY, Sheila; CRAMER, Tom. "What? So what": the next-generation JHOVE2 architecture for format-aware characterization. **International Journal of Digital Curation**, Edinburgh, v. 4, n. 3, p. 123-136, 2009.

ABRAMS, Stephen *et al.* Curation micro-services: a pipeline metaphor for repositories. **Journal of Digital Information**, *[s. l.]*, v. 12, n. 2, 2011. Disponível em: https://jodi-ojs-tdl.tdl.org/jodi/article/view/1605. Acesso em: 6 abr. 2026.

ACADEMIC PRESERVATION TRUST. **DART**: digital archivist's resource tool. Versão 3. Virginia: APTrust, 2026. Disponível em: https://aptrust.org/documentation-page/dart-digital-archivists-resource-tool/. Acesso em: 8 maio 2026.

ANDRADE, Nuno Emanuel Paiva de. **Políticas de preservação digital em bibliotecas universitárias**. 2024. Dissertação (Mestrado em Ciência da Informação) – Universidade do Porto, Porto, 2024. Disponível em: https://repositorio-aberto.up.pt/handle/10216/161343. Acesso em: 8 abr. 2026.

CALIFORNIA DIGITAL LIBRARY. **UC3 curation foundations**: rev. 0.13. *[S. l.: s. n.]*, 2010. 29 p. Disponível em: https://web.archive.org/web/20180520141130/https://confluence.ucop.edu/download/attachments/13860983/UC3-Foundations-latest.pdf. Acesso em: 29 abr. 2026.

CAPLAN, Priscilla. **Understanding PREMIS**: an overview of the PREMIS Data Dictionary for Preservation Metadata. Washington, DC: Library of Congress, 2009. Disponível em: https://www.loc.gov/standards/premis/understanding-premis.pdf. Acesso em: 11 abr. 2026.

CISCO SYSTEMS. **ClamAV documentation**. *[S. l.]*: Cisco Systems, 2024. Disponível em: https://docs.clamav.net. Acesso em: 19 abr. 2026.

CONSELHO NACIONAL DE ARQUIVOS (CONARQ) (Brasil). **Recomendações para digitalização de documentos arquivísticos permanentes**. Rio de Janeiro: CONARQ, 2010. Disponível em: https://www.gov.br/conarq/pt-br/centrais-de-conteudo/publicacoes. Acesso em: 8 abr. 2026.

CONSELHO NACIONAL DE ARQUIVOS (CONARQ) (Brasil**). Diretrizes para a implementação de repositórios arquivísticos digitais confiáveis (RDC-Arq)**. Rio de Janeiro: CONARQ, 2015.

CONSELHO NACIONAL DE ARQUIVOS (CONARQ) (Brasil). **Relatório técnico sobre censo de arquivos públicos instituídos no Brasil**. Rio de Janeiro: CONARQ, 2020a.

CONSELHO NACIONAL DE ARQUIVOS (CONARQ) (Brasil). **Glossário**: documentos arquivísticos digitais. Versão 8.0. Rio de Janeiro: CONARQ, 2020b.

CONSULTATIVE COMMITTEE FOR SPACE DATA SYSTEMS (CCSDS). **Reference model for an Open Archival Information System (OAIS)**: Magenta Book, Issue 2. Washington, DC: CCSDS, 2012. (CCSDS 650.0-M-2). Disponível em: http://urlib.net/sid.inpe.br/mtc-m18/2012/07.12.18.08. Acesso em: 8 abr. 2026.

CONSULTATIVE COMMITTEE FOR SPACE DATA SYSTEMS (CCSDS). **Reference model for an Open Archival Information System (OAIS)**: recommended practice. 3. ed. Washington, DC: CCSDS, 2024. (CCSDS 650.0-M-3). Disponível em: https://ccsds.org/publications/reference-models/. Acesso em: 7 abr. 2026.

DAPPERT, Angela; FARQUHAR, Adam. Modelling organizational preservation goals to guide digital preservation. **International Journal of Digital Curation**, Edinburgh, v. 4, n. 2, p. 119-134, 2009. Disponível em: https://ijdc.net/index.php/ijdc/article/view/123. Acesso em: 7 abr. 2026.

DILCIS BOARD. **E-ARK CSIP**: common specification for information packages. Versão 2.1. *[S. l.]*: DILCIS Board, 2021. Disponível em: https://earkcsip.dilcis.eu/pdf/eark-csip.pdf. Acesso em: 31 mar. 2026.

DURANTI, Luciana. Concepts, principles, and methods for the management of electronic records. **The Information Society**, *[s. l.]*, v. 17, n. 4, p. 271-279, 2001.

DURANTI, Luciana (ed.). **The long-term preservation of authentic electronic records**: findings of the InterPARES Project. San Miniato: Archilab, 2005.

DURANTI, Luciana; PRESTON, Randy (ed.). **International research on permanent authentic records in electronic systems (InterPARES) 2**: experiential, interactive and dynamic records. Padova: Associazione Nazionale Archivistica Italiana, 2008. Disponível em: http://www.interpares.ca/ip2/book.cfm. Acesso em: 6 maio 2026.

DURANTI, Luciana; ROGERS, Corinne (ed.). **Trusting records in the cloud**. London: DPC, 2019. Disponível em: https://www.dpconline.org/docs/miscellaneous/research/2591-interpares-trust-synthesis-report/file. Acesso em: 6 maio 2026.

EICAR. **The anti-malware testfile**. *[S. l.]*: EICAR, 2026. Disponível em: https://www.eicar.org/download-anti-malware-testfile/. Acesso em: 22 abr. 2026.

FARQUHAR, Adam; HOCKX-YU, Helen. Planets: integrated services for digital preservation. **International Journal of Digital Curation**, Edinburgh, v. 2, n. 2, p. 88-99, 2007. Disponível em: https://ijdc.net/index.php/ijdc/article/view/32. Acesso em: 8 abr. 2026.

FERREIRA, Miguel. **Introdução à preservação digital**: conceitos, estratégias e actuais consensos. Guimarães: Escola de Engenharia da Universidade do Minho, 2006. 88 p. Disponível em: https://repositorium.sdum.uminho.pt/bitstream/1822/5820/1/livro.pdf. Acesso em: 18 abr. 2026.

FFMPEG PROJECT. **FFmpeg documentation**. *[S. l.]*: FFmpeg Project, 2024. Disponível em: https://ffmpeg.org/documentation.html. Acesso em: 7 abr. 2026.

FREE SOFTWARE FOUNDATION. **rsync(1)**: Linux man page. *[S. l.]*: FSF, 2024. Disponível em: https://download.samba.org/pub/rsync/rsync.1. Acesso em: 20 abr. 2026.

GILLILAND, Anne J. Setting the stage. *In:* BACA, Murtha (org.). **Introduction to metadata**. 3. ed. Los Angeles: Getty Publications, 2016. p. 1-19.

GRÁCIO, José Carlos Abbud. **Preservação digital na gestão da informação:** um modelo processual para as instituições de ensino superior. [*S. l.*]: Cultura Acadêmica Editora, 2012.

HARVARD LIBRARY. **FITS**: file information tool set. *[S. l.]*: Harvard Library, 2023. Disponível em: https://projects.iq.harvard.edu/fits. Acesso em: 28 abr. 2026.

HARVEY, Phil. **ExifTool**: read, write and edit meta information. *[S. l.: s. n.]*, 2024. Disponível em: https://exiftool.org. Acesso em: 8 abr. 2026.

HIGGINS, Sarah. The DCC curation lifecycle model. **International Journal of Digital Curation**, Edinburgh, v. 3, n. 1, p. 134-140, 2008. Disponível em: https://ijdc.net/index.php/ijdc/article/view/69. Acesso em: 7 abr. 2026.

IMAGEMAGICK STUDIO LLC. **ImageMagick documentation**. *[S. l.]*: ImageMagick Studio, 2024. Disponível em: https://imagemagick.org/script/command-line-tools.php. Acesso em: 25 abr. 2026.

INTERPARES 3 PROJECT TEAM. **Preserver guidelines**: making and maintaining digital materials: guidelines for individuals. Vancouver: InterPARES 3, 2012. Disponível em: http://www.interpares.ca/ip3/display_file.cfm?doc=ip3_canada_guidance_note_1_v1.pdf. Acesso em: 2 abr. 2026.

JOINT INFORMATION SYSTEMS COMMITTEE. **JISC circular 6/03**. *[S. l.]*: JISC, 2003. Disponível em: https://web.archive.org/web/20060501153454/http://www.dcc.ac.uk:80/docs/6-03Circular.pdf. Acesso em: 11 abr. 2026.

KORNBLUM, Jesse. **hashdeep**: a cross-platform tool to compute hashes. *[S. l.: s. n.]*, 2013. Disponível em: https://github.com/jessek/hashdeep. Acesso em: 19 abr. 2026.

KUNZE, John *et al.* **The BagIt file packaging format (V1.0)**: RFC 8493. *[S. l.]*: IETF, 2018. Disponível em: https://www.rfc-editor.org/rfc/rfc8493. Acesso em: 19 abr. 2026.

LAVOIE, Brian. **The Open Archival Information System (OAIS) reference model**: introductory guide. 2. ed. York: Digital Preservation Coalition, 2014. (DPC Technology Watch Report, 14-02). Disponível em: https://dx.doi.org/10.7207/twr14-02. Acesso em: 12 abr. 2026.

LEHANE, Richard. **Siegfried**: a file format identification tool. *[S. l.: s. n.]*, 2014. Disponível em: https://www.itforarchivists.com/siegfried. Acesso em: 8 abr. 2026.

LEWIS, James; FOWLER, Martin. **Microservices:** a definition of this new architectural term. [S. l.], 25 mar. 2014. Disponível em: https://martinfowler.com/articles/microservices.html. Acesso em: 11 abr. 2026.

LIBRARY OF CONGRESS. **BagIt-Python**: a Python library for working with BagIt packages. *[S. l.]*: Library of Congress, 2022. Disponível em: https://github.com/LibraryOfCongress/bagit-python. Acesso em: 20 abr. 2026.

MÁRDERO ARELLANO, Miguel Ángel. **Preservação de documentos digitais**. Ciência da Informação, Brasília, v. 33, n. 2, p. 15-27, maio/ago. 2004. Disponível em: https://revista.ibict.br/ciinf/article/view/1043. Acesso em: 18 abr. 2026.

MEDIAAREA.NET. **MediaInfo documentation**. *[S. l.]*: MediaArea, 2024a. Disponível em: https://mediaarea.net/en/MediaInfo/Documentation. Acesso em: 23 abr. 2026.

MEDIAAREA.NET. **MediaConch documentation**. *[S. l.]*: MediaArea, 2024b. Disponível em: https://mediaarea.net/MediaConch. Acesso em: 23 abr. 2026.

NATIONAL ARCHIVES (Reino Unido). **DROID**: digital record object identification. *[S. l.]*: The National Archives, 2023. Disponível em: https://www.nationalarchives.gov.uk/information-management/manage-information/policy-process/digital-continuity/file-profiling-tool-droid/. Acesso em: 25 abr. 2026.

NATIONAL LIBRARY OF NEW ZEALAND. **Metadata extraction tool**. Versão 3.G. Wellington: NLNZ, 2007. Disponível em: http://meta-extractor.sourceforge.net/. Acesso em: 27 abr. 2026.

NDSA LEVELS OF PRESERVATION WORKING GROUP. **Levels of digital preservation, version 2.0**. Washington, DC: National Digital Stewardship Alliance, 2019. Disponível em: https://ndsa.org/publications/levels-of-digital-preservation/. Acesso em: 20 abr. 2026.

OPEN PRESERVATION FOUNDATION. **JHOVE**: JSTOR/Harvard object validation environment. *[S. l.]*: Open Preservation Foundation, 2024. Disponível em: https://jhove.openpreservation.org. Acesso em: 11 abr. 2026.

OPEN PRESERVATION FOUNDATION. **format-corpus**. *[S. l.]*: Open Preservation Foundation, 2026 [2012]. Disponível em: https://github.com/openpreserve/format-corpus. Acesso em: 12 abr. 2026.

PAVÃO, Caterina G.; CAREGNATO, Sônia E.; ROCHA, Rafael Port da. Implementação da preservação digital em repositórios: conhecimento e práticas. **RDBCI: Revista Digital de Biblioteconomia e Ciência da Informação**, *[s. l.]*, v. 14, n. 3, p. 407-426, 2016.

PDF ASSOCIATION. **veraPDF**: industry supported PDF/A validation. *[S. l.]*: PDF Association, 2024. Disponível em: https://verapdf.org. Acesso em: 28 abr. 2026.

PREMIS EDITORIAL COMMITTEE. **PREMIS data dictionary for preservation metadata**: version 3.0. Washington, DC: Library of Congress, 2015. Disponível em: https://www.loc.gov/standards/premis/v3/premis-3-0-final.pdf. Acesso em: 18 abr. 2026.

RCLONE. **Rclone documentation**. *[S. l.]*: Rclone, 2024. Disponível em: https://rclone.org/docs/. Acesso em: 29 abr. 2026.

RESEARCH LIBRARIES GROUP; ONLINE COMPUTER LIBRARY CENTER. **Trusted digital repositories**: attributes and responsibilities. Mountain View: RLG, 2002.

ROCHA, Rafael Port da. **Corpo de conhecimento em curadoria digital**: Fabrico. Porto Alegre: Projeto Fabrico, 2025a. Disponível em: https://www.ufrgs.br/fabrico/curadoriadigital.php. Acesso em: 8 abr. 2026.

ROCHA, Rafael Port da. Digitalização e preservação digital: abordagens metodológicas. In: SILVA, Fabiano Couto Corrêa da; BARROS, Thiago Henrique Bragato (org.). **Caminhos Metodológicos em Ciência da Informação**: Fundamentos e Aplicações. São Paulo: Pimenta Cultural, 2025b.

SAYÃO, Luís Fernando. Uma outra face dos metadados: informações para a gestão da preservação digital. **Encontros Bibli: revista eletrônica de biblioteconomia e ciência da informação**, Florianópolis, v. 15, n. 30, p. 1-31, 2010. DOI: 10.5007/1518-2924.2010v15n30p1. Disponível em: https://periodicos.ufsc.br/index.php/eb/article/view/1518-2924.2010v15n30p1. Acesso em: 6 abr. 2026.

SCHMIDT JÚNIOR, Norman. **TCC-preservacao-instituicoes-pequeno-porte**: scripts Python para o workflow de preservação digital em instituições de memória de pequeno porte. [S. l.]: Zenodo, 2026. Disponível em: https://doi.org/10.5281/zenodo.20636201. Acesso em: 11 jun. 2026.

THE DOCUMENT FOUNDATION. **LibreOffice documentation**. *[S. l.]*: The Document Foundation, 2024. Disponível em: https://documentation.libreoffice.org. Acesso em: 28 abr. 2026.

THOMAZ, Katia P.; SOARES, Antonio José. A preservação digital e o modelo de referência Open Archival Information System (OAIS). **DataGramaZero: Revista de Ciência da Informação**, Rio de Janeiro, v. 5, n. 1, 2004. Disponível em: https://brapci.inf.br/index.php/res/download/45229. Acesso em: 30 abr. 2026.

VAN GARDEREN, Peter. Archivematica: using micro-services and open-source software to deliver a comprehensive digital curation solution. *In:* INTERNATIONAL CONFERENCE ON PRESERVATION OF DIGITAL OBJECTS (iPRES), 7., 2010, Viena. **Proceedings [...]**. Viena, 2010.

WATERS, Donald; GARRETT, John. **Preserving digital information**: report of the Task Force on Archiving of Digital Information. Washington, DC: Commission on Preservation and Access; RLG, 1996. Disponível em: https://www.clir.org/wp-content/uploads/sites/6/pub63watersgarrett.pdf. Acesso em: 1 abr. 2026.

APÊNDICE A — REPOSITÓRIO DE SCRIPTS DO WORKFLOW

Os scripts Python que implementam o orquestrador e os microsserviços MS1–MS9 do workflow de preservação digital proposto neste trabalho estão disponíveis em repositório público no GitHub, arquivado com identificador persistente (DOI) no Zenodo:

Repositório: https://github.com/normanschmidt/TCC-preservacao-instituicoes-pequeno-porte

DOI (versão 1.0.0): https://doi.org/10.5281/zenodo.20636201

Licença: GNU General Public License v3.0 (GPL-3.0)

O quadro a seguir relaciona cada script ao microsserviço e às etapas do workflow correspondentes, conforme definido nas seções 5 e 6, e descreve outros arquivos não diretamente usados no workflow, porém presentes no repositório.

Quadro — Estrutura do repositório

```
TCC-preservacao-instituicoes-pequeno-porte/
├── orquestrador.py           Orquestrador de execução das etapas E1–E9
├── admissao.py               MS1 — Admissão de conteúdo (E2)
├── identificacao.py          MS2 — Identificação de formato (E4)
├── conformidade.py           MS3 — Validação de conformidade (E5)
├── caracterizacao.py         MS4 — Caracterização técnica (E6)
├── verificacao.py            MS5 — Verificação de integridade (E10)
├── empacotamento.py          MS6 — Empacotamento BagIt (E3, E8)
├── eventos.py                MS7 — Registro de eventos PREMIS
├── replicacao.py             MS8 — Replicação para destinos (E9)
├── derivadas.py              MS9 — Geração de rep. derivadas (E7)
├── destinos.exemplo.json     Modelo de conf. destinos de replicação
├── comum.py                  Funções auxiliares compartilhadas
├── validar_premis.py         Auxiliar validação PREMIS contra XSD
├── premis.xsd                Esquema XSD 3.0 PREMIS (2016)
├── metadados_descritivos.py  Auxiliar validação metadados
├── requirements.txt          Dependências Python do projeto
├── README.md                 Descrição geral e instruções de uso
├── INSTALACAO.md             Guia de instalação do ambiente
├── CITATION.cff              Metadados de citação do software
├── LICENSE                   Licença GNU General Public License v3.0
└── .gitignore                Exclusões do controle de versão
```

Fonte: Elaborado pelo autor

Instruções de instalação e uso: ver README.md e INSTALACAO.md no repositório.

Dependências de execução: listadas em requirements.txt no repositório.