Brasília | dezembro de 2019

Relatório parcial

IBICT | TJDFT

Projeto

Estudo para Implantação de Repositório Arquivístico Digital Confiável – RDC–Arq no âmbito do Tribunal de Justiça do Distrito Federal e Territórios-TJDFT

Elaboração

Tiago Emmanuel Nunes Braga
Alexandre Faria de Oliveira
Miguel Angel Madero Arellano
Marcos Pereira de Novais
Ana Clara Fonseca Vassallo
Luísa Kruchak Barros
Tatiana Canelhas Pignataro
Maxwell Borges Bezerra
Henrique Oliveira Costa

Relatório parcial relativo à Meta 3: implementação

ibict 65 | MINISTÉRIO DA CIÊNCIA, TECNOLOGIA, INOVAÇÕES E COMUNICAÇÕES | TJDFT | Poder Judiciário da União Tribunal de Justiça do Distrito Federal e dos Territórios | PÁTRIA AMADA BRASIL GOVERNO FEDERAL

Instituto Brasileiro de Informação em Ciência e Tecnologia

Diretora
Cecilia Leite Oliveira
Coordenação-Geral de Pesquisa e Desenvolvimento de Novos Produtos
Marcel Garcia de Souza (substituto)
Coordenação-Geral de Pesquisa e Manutenção de Produtos Consolidados
Bianca Amaro de Melo
Coordenação-Geral de Tecnologias de Informação e Informática
Tiago Emmanuel Nunes Braga

Sobre esse documento
Responsável pelo relatório
Tiago Emmanuel Nunes Braga
Apoio na elaboração
Alexandre Faria de Oliveira
Miguel Angel Madero Arellano
Marcos Pereira de Novais
Ana Clara Fonseca Vassallo
Luísa Kruchak Barros
Tatiana Canelhas Pignataro
Maxwell Borges Bezerra
Henrique Oliveira Costa

Contato
tiagobraga@ibict.br | (61) 3217.6485

Relatório parcial

Elaboração

Tiago Emmanuel Nunes Braga

Alexandre Faria de Oliveira

Miguel Angel Madero Arellano

Marcos Pereira de Novais

Ana Clara Fonseca Vassallo

Luísa Kruchak Barros

Tatiana Canelhas Pignataro

Maxwell Borges Bezerra

Henrique Oliveira Costa

Setor de Autarquias (SAUS) Quadra 5 Bloco H Lote 6 – CEP: 70070-912 – Brasília – DF

Tel.: +55 (61) 3217.6350 | www.ibict.br

Setor de Autarquias (SAUS) Quadra 5 Bloco H Lote 6 – CEP: 70070-912 – Brasília – DF

Tel.: +55 (61) 3217.6350 | www.ibict.br

Sumário

Lista de Figuras

Introdução

O projeto estudo para implantação de repositório arquivístico digital confiável – RDC – Arq no âmbito do Tribunal de Justiça do Distrito Federal e Territórios - TJDFT tem como foco a execução de quatro metas:

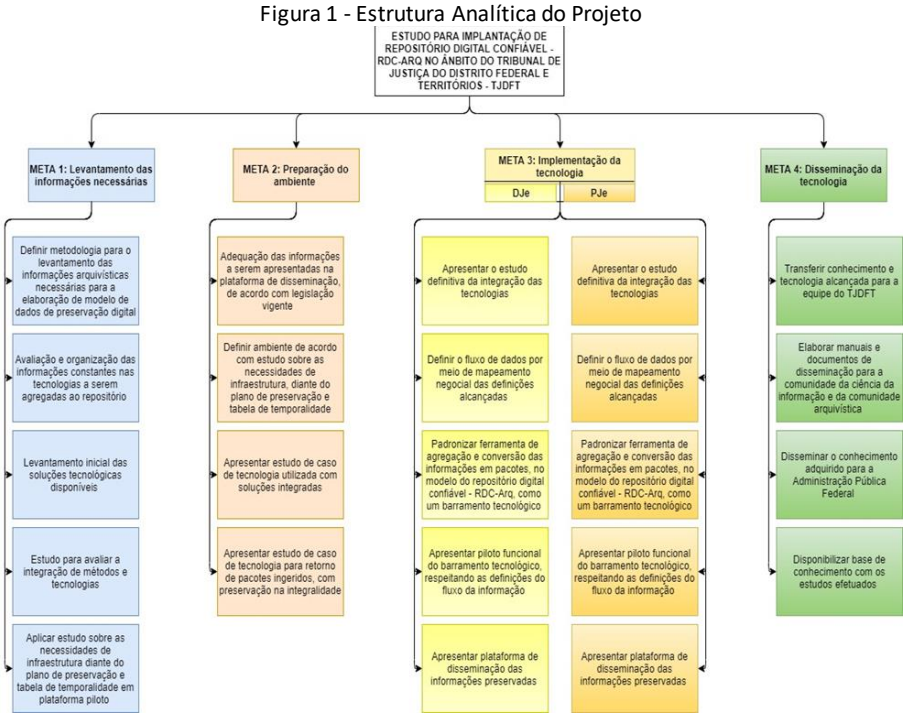
- Levantamento de informações necessárias;
- Preparação de ambiente;
- Implementação da tecnologia; e
- Disseminação da tecnologia.

Este relatório refere-se à conclusão parcial da Meta 3, Implementação da Tecnologia. Esta meta baseou-se na implementação dos artefatos tecnológicos definidos e estruturados nas duas metas anteriores, levantamento de informações necessárias e preparação do ambiente, relativos ao Diário de Justiça eletrônico - DJe.

Durante esta etapa da pesquisa foram realizadas as seguintes ações:

- Integração de tecnologias;
- Definição do fluxo de dados por meio de mapeamento negocial;
- Padronização do barramento gerador de pacotes para RDC-Arq;
- Apresentação do piloto funcional; e
- Apresentação da plataforma de disseminação das informações preservadas.

Figura 1 - Estrutura Analítica do Projeto



Fonte: elaboração própria

Uma observação que se faz pertinente é a alteração do escopo da meta 3, que foi desmembrada em duas etapas: a primeira, o desenvolvimento completo do barramento capaz de realizar a integração DJe / Archivematica; e a segunda, a integração do PJe / Archivematica. Esta alteração ocorreu por uma série de

razões, dentre as quais, mudança do escopo tecnológico da integração, com adição de processos estabelecidos pelo CNJ, tais como a utilização do Modelo Nacional de Interoperabilidade (MNI).

A seguir no relatório serão descritas as ações realizadas durante a execução da meta 3 correspondentes à primeira etapa, bem como os principais resultados obtidos. Todo o desenvolvimento das ações ora descritas foi acompanhado de maneira próxima pela equipe do TJDFT, e validadas nas reuniões de acompanhamento.

A integração tecnológica

Para a criação da integração tecnológica, foi necessário estudar, propor e apresentar a integração ocorrida entre o sistema DJe e o Archivematica por meio do barramento. Para isso, foi necessário estabelecer o ambiente adequado e levantar informações junto aos especialistas de cada ferramenta. O ambiente estudado resultou na instalação e configuração do barramento na máquina do próprio Archivematica, aproveitando a estrutura de acesso e pasta da instalação já encontrada no servidor.

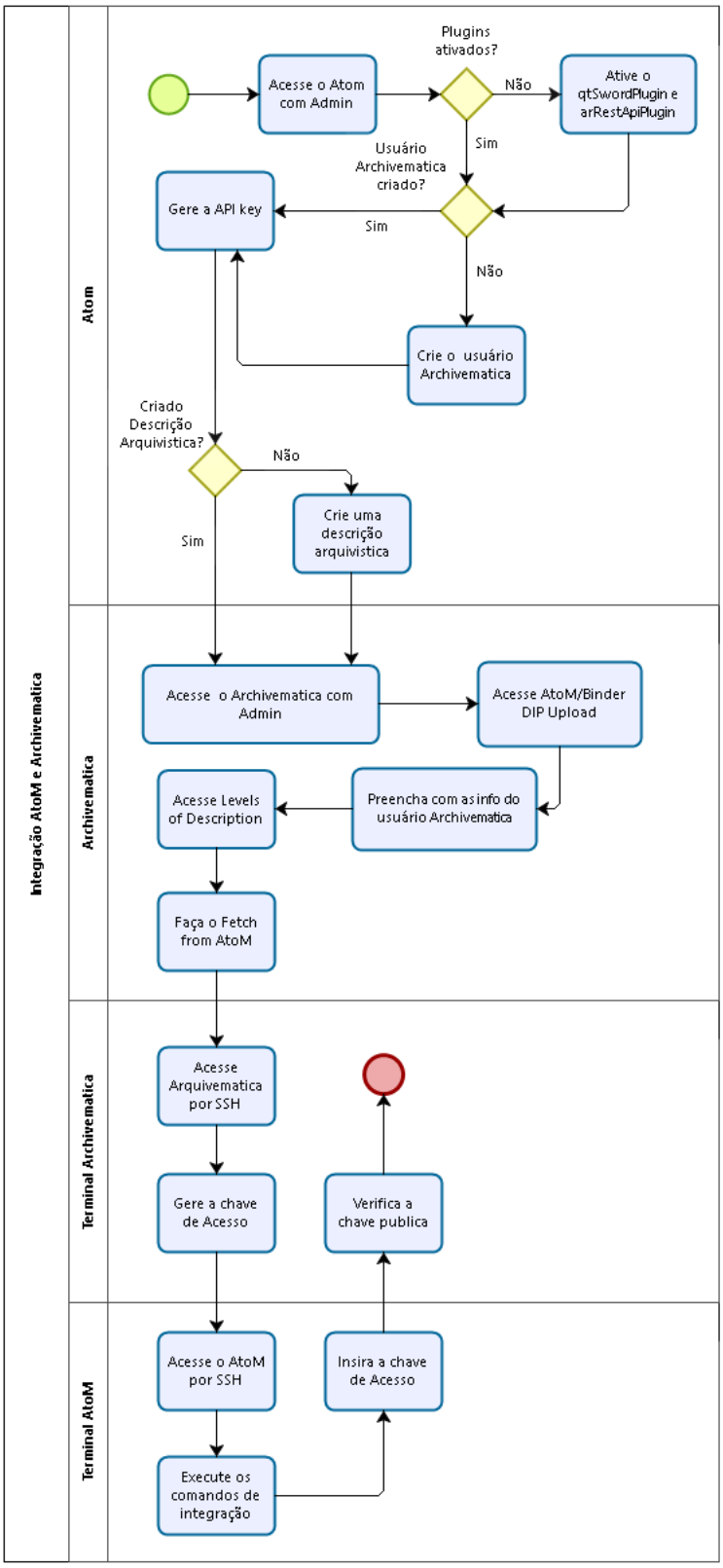
Neste nível, o barramento foi ponto chave para a integração de comandos, por meio de utilização do método REST, já utilizado pelo Archivematica. Como modelo de integração foram estudadas as tecnologias vigentes e necessárias para a disponibilização do RDC-Arq. Este estudo determinou a utilização de certas ferramentas como o caso do ATOM e fortemente a linguagem de programação PYTHON.

Para a devida conexão entre os sistemas, foi necessária a liberação de acesso de máquina, entre o Archivematica e o Barramento e os servidores de aplicação, banco de dados e arquivos em pasta, de forma segura e interna, sem quebra de comunicação, com segurança técnica aplicada, caminhos e meios determinados e com acesso pelo barramento de somente leitura.

Entre outras informações, foi possível estudar e determinar que cada aplicação deve disponibilizar um usuário de consulta para o barramento, permitindo que as conexões sejam autênticas e sem interferência de retorno para os dados originais.

Em fase de apresentação, foi necessário estabelecer o Atom como ferramenta de disseminação, instalada e disponibilizada em servidor com acesso externo, destacado da instalação do Archivematica e utilizando a metodologia SWORD para receber os depósitos advindos do RDC-Arq. Houve a necessidade de confirmação e identificação mais fortemente amarrada ao ID, conforme demonstrado no diagrama a seguir.

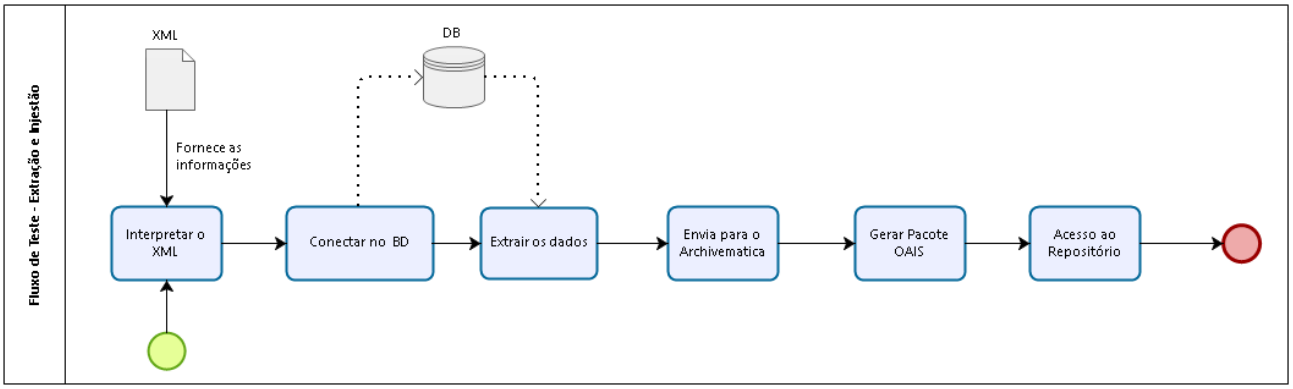
Figura 2 - Integração AtoM e Archivematica



Fonte: elaboração própria

Para que as ações pudessem ser assumidas pelo barramento foi preciso estabelecer fluxo de teste capaz de abranger a necessidade do projeto. Este fluxo contemplava as ações de extração e ingestão dos dados para que o pacote OAIS fosse gerado pelo Archivematica.

Figura 3 - Fluxo de teste

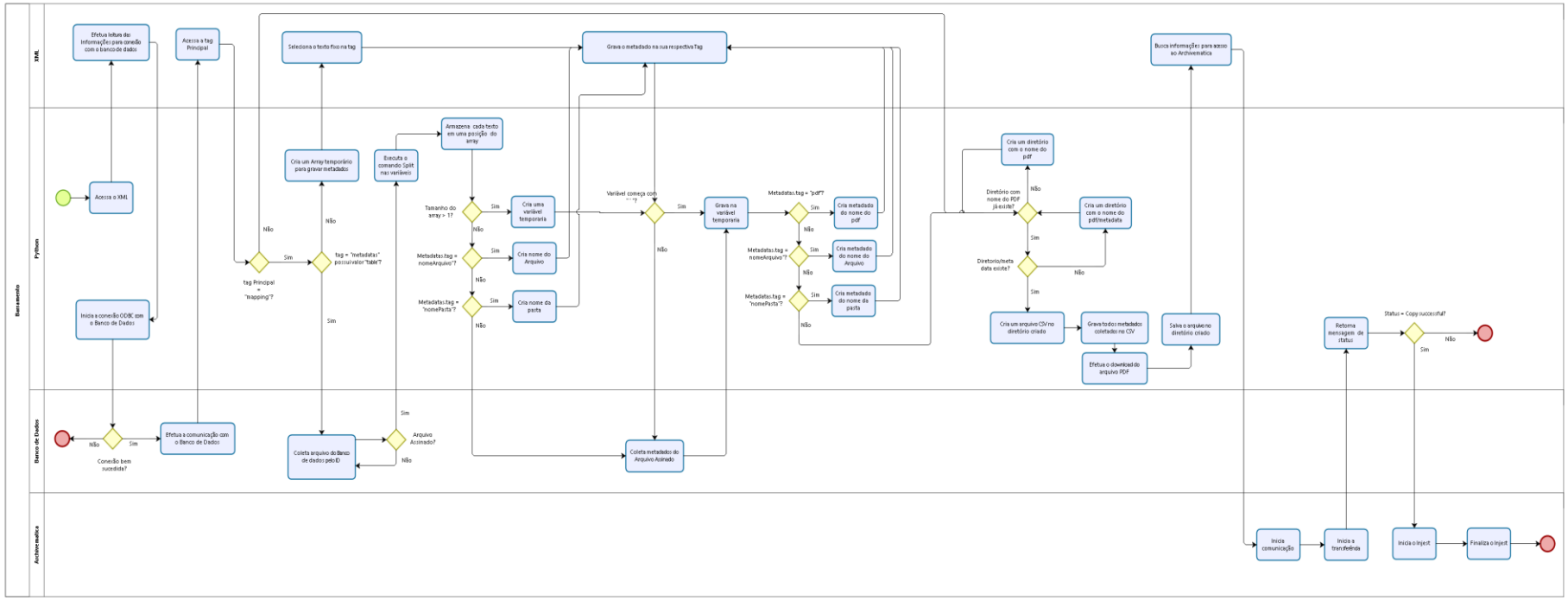


Fonte: elaboração própria

Neste fluxo foi possível testar e estabelecer a melhora do XML, que teve sua estrutura definida para mapear o sistema DJe. A mesma estrutura será utilizada para mapear o sistema PJe, em fase de construção. Alguns dados não serão detalhados no relatório, visando a segurança da informação, no entanto são contemplados os conteúdos essenciais, como a estrutura do XML resultante da conexão e definição de tradução de dados em metadados em ISAD-G e DUBLIN CORE, conforme já definido em relatório anterior.

Para o fluxo de estudo e estabelecimento do fluxo completo de transação, foi necessário a descrição detalhada dos processos a serem seguidos, conforme é possível observar no diagrama a seguir.

Figura 4 - Fluxo barramento



Fonte: elaboração própria

Em sua primeira parte, o XML foi escrito com as definições e dados de conexão. Para a abertura da sua informação é necessário o entendimento de que se trata de um XML ativo, de característica própria e genérica, sendo possível utilizá-lo e expandi-lo de forma a atender outros sistemas a serem integrados.

Aqui, seguem as informações contidas em sua primeira parte, que contém os dados de conexão e identificação de rede, conforme já estabelecido em relatório anterior e com as definições aprimoradas para o funcionamento real junto ao DJe.

```
<definition>
    <conections>
        <driver>InterSystems ODBC</driver>
        <server>127.0.0.1</server>
        <port>3306</port>
        <database>DATABASE</database>
        <uid>USER</uid>
        <pwd>PASSWORD</pwd>
</conections>
```

Estabelecendo assim, os dados de conexão do banco, sua porta, usuário e senha a serem apresentados para esta conexão. Em seguida, foi mapeado o sistema de arquivos que depois foram copiados e disponibilizados para o Archivematica.

Nesta segunda parte o código utilizado é apresentado a seguir:

```
    <fileserver>
        <server>127.0.0.1</server>
        <port>80</port>
        <user>USER</user>
        <api_key>API_KEY</api_key>
        <folder_uuid>FOLDER_UUID</folder_uuid>
        <package_type>standard</package_type>
    </fileserver>
```

As informações de usuário e chave de acesso, bem como a identificação da pasta, são necessárias para que sejam copiados com sucesso os dados representados no banco de dados, com a funcionalidade preservada e demais interações, sem a permissão de escrita em pasta, sendo somente leitura.

Nas seções a seguir, são apresentadas as demais estruturas de dados e representação em metadados dos itens coletados pelo barramento, neste sentido, optou-se pela demonstração da origem dos dados e sua devida escrita em arquivo próprio de reconhecimento do Archivematica, já retratado em relatórios anteriores.

Primeiro conjunto de temporalidade e montagem de conjunto de metadados:

```
<mapping>
 <temporality field="DiarioEletronico.DataPublicacao">+2</temporality>
 <filename format="object/"></filename>
 <relation table="DiarioEletronico">
  <reference column="ID">DatasDePublicacao.ID</reference>
```

```
 </relation>

  <metadatas table="tjdf_dje.DiarioEletronico">
    <nomePasta>'dje_' Ano '_' Numero</nomePasta>
    <nomeArquivo>'DJ' Numero '_' Ano</nomeArquivo>
    <filename>'objects/DJ' Numero '_' Ano '.pdf'</filename>
    <numeroSistema>ID</numeroSistema>
    <numeroIdentificador>Ano Numero</numeroIdentificador>
    <responsavelPubLogin>Login</responsavelPubLogin>
    <responsavelRedator>Login</responsavelRedator>
    <dataCriacao>DataHoraDeModificacao</dataCriacao>
    <dataPublicacao>DataDePublicacao</dataPublicacao>
    <pdf>Ano Numero</pdf>
    <anoDiario>Ano</anoDiario>
    <numeroDiario>Numero</numeroDiario>
    <dc.title>Numero Ano</dc.title>
    <dc.date>DataDePublicacao</dc.date>
    <dc.identifier>Ano Numero</dc.identifier>
  </metadatas>
```

Em seguida, os demais dados:

```
            <codigoClassificacao>06.04.01</codigoClassificacao>
            <assuntoClassificacao>Gestão de documentos e informações > Produção
editorial > Edição, co-edição e folder > Periódicos</assuntoClassificacao>
            <versaoPCTT>Plano de Classificação e Tabela de Temporalidade de
Documentos - PCTT - AD TJDFT (Minuta de Proposta do PCTT-TJDFT - Versão 1-
47)</versaoPCTT>
            <prazoGuardaClassificacao>1 ano de guarda em arquivo corrente e depois
é recolhido para o arquivo permanente.</prazoGuardaClassificacao>
            <destinacaoPrevista>Repositório Arquivístico Digital - Tribunal de
Justiça do Distrito Federal e dos Territórios</destinacaoPrevista>
            <grauSigilo>Acesso integral</grauSigilo>
            <restricaoAcesso>Sem restrição</restricaoAcesso>
            <meioDocumental>Documento digital</meioDocumental>
            <generoDocumental>Textual</generoDocumental>
            <especieDocumental>Diário da Justiça</especieDocumental>
            <tipoDocumental>Diário de justiça para publicação diária de atos
judiciais e administrativos e informações relevantes à esfera do TJDFT.
</tipoDocumental>
            <localizacao>Documento encontrado na plataforma online do Diário da
Justiça do TJDFT.</localizacao>
            <procedencia>NUGAD - Núcleo de Gestão de Sistemas
Administrativos</procedencia>
            <assinaturaDigital></assinaturaDigital>
            <algoritimoChecksum>MD5</algoritimoChecksum>
            <isad.identifier></isad.identifier>
            <isad.title></isad.title>
            <isad.eventDates></isad.eventDates>
            <isad.levelOfDescription></isad.levelOfDescription>
            <isad.extentAndMedium></isad.extentAndMedium>
            <isad.eventActors>Tribunal de Justiça do Distrito Federal e dos
Territórios</isad.eventActors>
            <isad.eventActorHistory></isad.eventActorHistory>
            <isad.archivalHistory></isad.archivalHistory>
            <isad.acquisition>Tribunal de Justiça do Distrito Federal e dos
Territórios</isad.acquisition>
            <isad.scopeAndContent></isad.scopeAndContent>
            <isad.appraisal>06.04.01 - Gestão de documentos e informações >
Produção editorial > Edição, co-edição e folder > Periódicos - 1 ano de guarda em
```

```
arquivo corrente e depois é recolhido para o arquivo permanente.</isad.appraisal>
            <isad.accruals></isad.accruals>
            <isad.arrangement></isad.arrangement>
            <isad.accessConditions></isad.accessConditions>
            <isad.reproductionConditions>Documento pode ser adquirido no site do
Diário da Justiça do TJDFT</isad.reproductionConditions>
            <isad.language>pt_BR</isad.language>

<isad.physicalCharacteristics>application/pdf</isad.physicalCharacteristics>
            <isad.findingAids></isad.findingAids>
            <isad.locationOfOriginals></isad.locationOfOriginals>
            <isad.locationOfCopies></isad.locationOfCopies>
            <isad.relatedUnitsOfDescription></isad.relatedUnitsOfDescription>
            <isad.publicationNote></isad.publicationNote>
            <isad.generalNote></isad.generalNote>
            <isad.archivistNote></isad.archivistNote>
            <isad.rules>Documento baseado nas regras da ISAD(G) do Conselho
Internacional de Arquivos</isad.rules>
            <isad.revisionHistory></isad.revisionHistory>
            <isad.placeAccessPoints></isad.placeAccessPoints>
            <isad.subjectAccessPoints></isad.subjectAccessPoints>
            <isad.genreAccessPoints>Textual</isad.genreAccessPoints>
            <isad.repository></isad.repository>
            <isad.script>latim</isad.script>
            <isad.languageNote></isad.languageNote>
            <isad.digitalObjectURI></isad.digitalObjectURI>
            <isad.digitalObjectChecksun></isad.digitalObjectChecksun>
            <isad.descriptionIdentifier></isad.descriptionIdentifier>
            <isad.institutionIdentifier></isad.institutionIdentifier>
            <isad.descriptionStatus>Draft</isad.descriptionStatus>
            <isad.levelOfDetail>Minimal</isad.levelOfDetail>
            <isad.languageOfDescription>pt_BR</isad.languageOfDescription>
            <isad.scriptOfDescription>latim</isad.scriptOfDescription>
            <isad.sources>https://pesquisadje.tjdft.jus.br/</isad.sources>
            <isad.publicationStatus>Draft</isad.publicationStatus>
            <isad.pyshicalObjectName></isad.pyshicalObjectName>
            <isad.physicalObjectLocation></isad.physicalObjectLocation>
            <isad.physicalObjectType></isad.physicalObjectType>
            <isad.alternativeIdentifiers></isad.alternativeIdentifiers>

<isad.alternativeIdentifiersLabels></isad.alternativeIdentifiersLabels>
            <isad.eventTypes></isad.eventTypes>
            <isad.eventStartDates></isad.eventStartDates>
            <isad.eventEndDates></isad.eventEndDates>
            <isad.eventDescriptions></isad.eventDescriptions>
            <isad.eventPlaces></isad.eventPlaces>
            <isad.culture></isad.culture>
            <dc.creator>Tribunal de Justiça do Distrito Federal e dos
Territórios</dc.creator>
            <dc.subject>assuntos cobertos por aquele diário</dc.subject>
            <dc.description></dc.description>
            <dc.publisher>Tribunal de Justiça do Distrito Federal e dos Territórios
</dc.publisher>
            <dc.constributor>Tribunal de Justiça do Distrito Federal e dos
Territórios</dc.constributor>
            <dc.type>texto</dc.type>
            <dc.format>application/PDF</dc.format>
            <dc.source>https://pesquisadje.tjdft.jus.br/</dc.source>
            <dc.language>pt_BR</dc.language>
            <dc.relation></dc.relation>
```

```
            <dc.coverage>Distrito Federal</dc.coverage>
            <dc.rights>Documento adquirido neste site, ou no site do Diário da
Justiça do Tribunal de Justiça do Distrito Federal e dos Territórios</dc.rights>
```

Finalizando com novos dados representados por consulta, uma vez que o DJe mantém seus dados em banco de dados junto com o PDF, permitindo assim o devido controle do que está em conformidade com a legislação e publicação vigente.

Os dados aqui apresentados devem ser reavaliados para o momento de divulgação, não sendo interessante aplicar dados e nome finais em documentos de disseminação de conhecimento.

```
<submitionDocumentation>
            <IndiceDaNoticia>SELECT      DISTINCT      IndiceDaNoticia.*      from
tjdf_dje.IndiceDaNoticia,tjdf_dje.DatasDePublicacao                          where
IndiceDaNoticia.TextoDaNoticia=DatasDePublicacao.TextoDaNoticia                and
DatasDePublicacao.DiarioEletronico=$@numeroDiario$                             and
YEAR(to_date(DatasDePublicacao.Data,                                        "yyyy-mm-
dd"))='$@anoDiario$';</IndiceDaNoticia>
            <CabecalhoDaNoticia>SELECT    DISTINCT    CabecalhoDaNoticia.*    from
tjdf_dje.CabecalhoDaNoticia,tjdf_dje.DatasDePublicacao                       where
CabecalhoDaNoticia.TextoDaNoticia=DatasDePublicacao.TextoDaNoticia             and
DatasDePublicacao.DiarioEletronico=$@numeroDiario$                             and
YEAR(to_date(DatasDePublicacao.Data,                                        "yyyy-mm-
dd"))='$@anoDiario$';</CabecalhoDaNoticia>
        </submitionDocumentation>
    </mapping>
</definition>
```

Na estrutura do XML finalizada para o sistema DJe foi notada a necessidade de se ter avaliação e evolução do XML em cada interação com os sistemas a serem integrados, demonstrando a capacidade de evolução do código do barramento. Porém, mesmo em casos mais extremos, a reconstrução do XML não é necessária, apenas modificação e ampliação de algumas informações, mantendo em grande contexto o que se necessita de cada um dos sistemas a serem coletados e preservados.

TRANSFERÊNCIAS PARA O ARCHIVEMATICA

São descritas as transferências de conteúdo para o Archivematica.

TRANSFERÊNCIAS DE BAIXA COMPLEXIDADE

Com exceção das transferências com bags compactadas, o Archivematica exige que todos os materiais na transferência estejam contidos em um diretório de nível superior. A estrutura de diretórios da transferência pode ser simples (ou seja, todos os arquivos localizados no mesmo diretório) ou pode estar em pastas e ser hierárquica.

No estudo feito para o DJe usaremos a transferência padronizada no Archivematica com apenas um diretório superior. A imagem a seguir mostra uma transferência básica chamada basicTransfer, onde dois objetos digitais estão no diretório de nível superior, enquanto mais dois objetos estão dentro de um subdiretório.

Figura 5 - Pacote na transferência de baixa complexidade

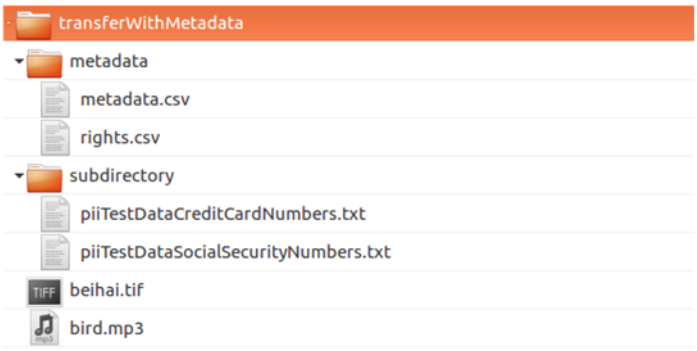
Fonte: Archivematica

Uma transferência pode ter quantos subdiretórios se desejar, embora isso possa afetar o tempo de processamento. Os objetos digitais podem estar em qualquer formato, apesar das diferentes possibilidades que o Archivematica possa operar com um determinado formato de arquivo. Esta é a estrutura básica de uma transferência no Archivematica; no entanto, existem muitas variações.

TRANSFERÊNCIAS COM METADADOS DESCRITIVOS E/OU DE DIREITOS

Para incluir metadados descritivos (que devem ser salvos em um documento chamado metadata, no formato CSV, com a codificação UTF8) e/ou metadados de direitos (que devem ser salvos em um documento chamado rights, no formato CSV, com a codificação UTF8) na transferência, deve-se adicionar um subdiretório também chamado “metadata”, localizado nível superior do pacote de transferência. A inclusão do diretório de metadados assegura que seus arquivos de metadados sejam marcados adequadamente como tal no arquivo METS.

Figura 6 - Pacote na transferência com metadados descritivos e de direitos



Fonte: Archivematica

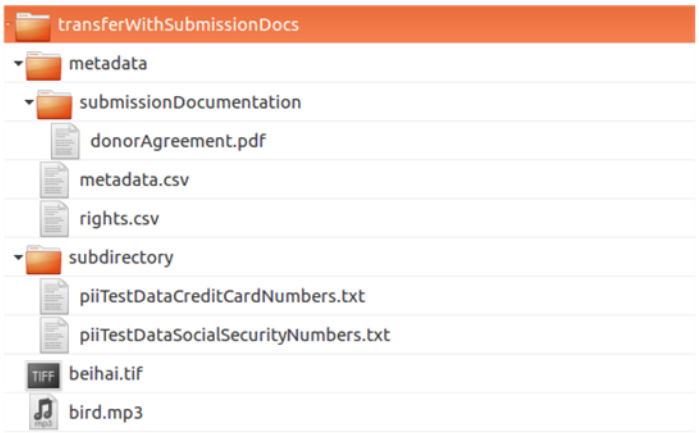
Os metadados inseridos no projeto serão de gestão, definidos pelo e-Arq Brasil e MoReq-Jus e metadados descritivos Dublin Core e ISAD(g). Serão extraídas informações do banco de dados dos sistemas do DJe e salvas no CSV. Mas nem todos os dados serão armazenados nesse arquivo, e isso será melhor explicado na documentação de submissão.

TRANSFERÊNCIAS COM DOCUMENTAÇÃO DE SUBMISSÃO

A documentação de submissão é um conceito no Archivematica que contabiliza materiais relacionados aos objetos digitais que estão sendo preservados, mas que não fazem parte estritamente da coleção – por exemplo, acordos de doadores, correspondência sobre materiais, relatórios de conservação etc. Se o Archivematica visualizar que uma transferência inclui documentação de submissão, pode incluir descrições deste material no arquivo AIP METS .

Para criar uma transferência que inclua documentação de submissão, o diretório de nível superior deve conter um subdiretório denominado *metadata*. Dentro do subdiretório *metadata* deve haver um diretório chamado *submissionDocumentation* contendo os arquivos de documentação de submissão.

Figura 7 - Pacote na transferência com metadados de submissão



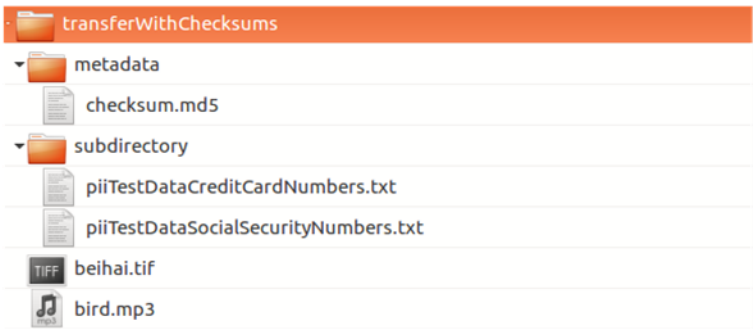
Fonte: Archivematica

No projeto, considera-se colocar, dentro desta pasta, dados sobre as partes processuais envolvidas no DJe. As tabelas consideradas importantes para serem adicionadas a esse diretório são *CabecalhoDaNoticia* e *IndiceDaNoticia* do banco (*tjdf_dje*), que representam as partes envolvidas em geral (relator, revisor, apelantes, advogados, impetrantes etc.), origem, despacho, número dos processos e outros dados relevantes relacionados às notícias publicadas.

TRANSFERÊNCIA COM CHECKSUMS

O Archivematica pode conferir os checksums (somas de verificação) MD5, SHA1, SHA256 e SHA512 que foram criados fora do sistema. Criar checksums fora do Archivematica é necessário se houver alguma preocupação com a perda de integridade dos dados durante a migração para o Archivematica, como no caso desse projeto, em que os objetos são extraídos pelo barramento e salvos em um diretório temporário até a concretização da transferência para o Archivematica. Os checksums são verificados durante o microsserviço "Verify transfer checksums" na guia Transfer. Os arquivos de checksums são colocados no diretório de metadados.

Figura 8 - Pacote na transferência com checksums



Fonte: Archivematica

Os arquivos de soma de verificação devem ter o seguinte nome:

- checksum.md5,
- checksum.sha1,
- checksum.sha256, ou
- checksum.sha512.

O próprio arquivo de checksum deve conter uma linha para cada soma de verificação, começando com o checksum, seguido por dois espaços e pelo caminho do arquivo, conforme exemplos abaixo:

```
2121dca88ad7f701d3f3e2d041004a56  beihai.tif
7f42199657dea535b6ad1963a6c7a2ac  bird.mp3
75388a532283b988f79206d63f65e9a2  subdirectory/piiTestDataCreditCardNumbers.txt
1d7193ea3b2193c79f55ea7e645503a9
subdirectory/piiTestDataSocialSecurityNumbers.txt
```

Se o checksum falhar, o microsserviço "Verify transfer checksums" mostrará um erro e a transferência é

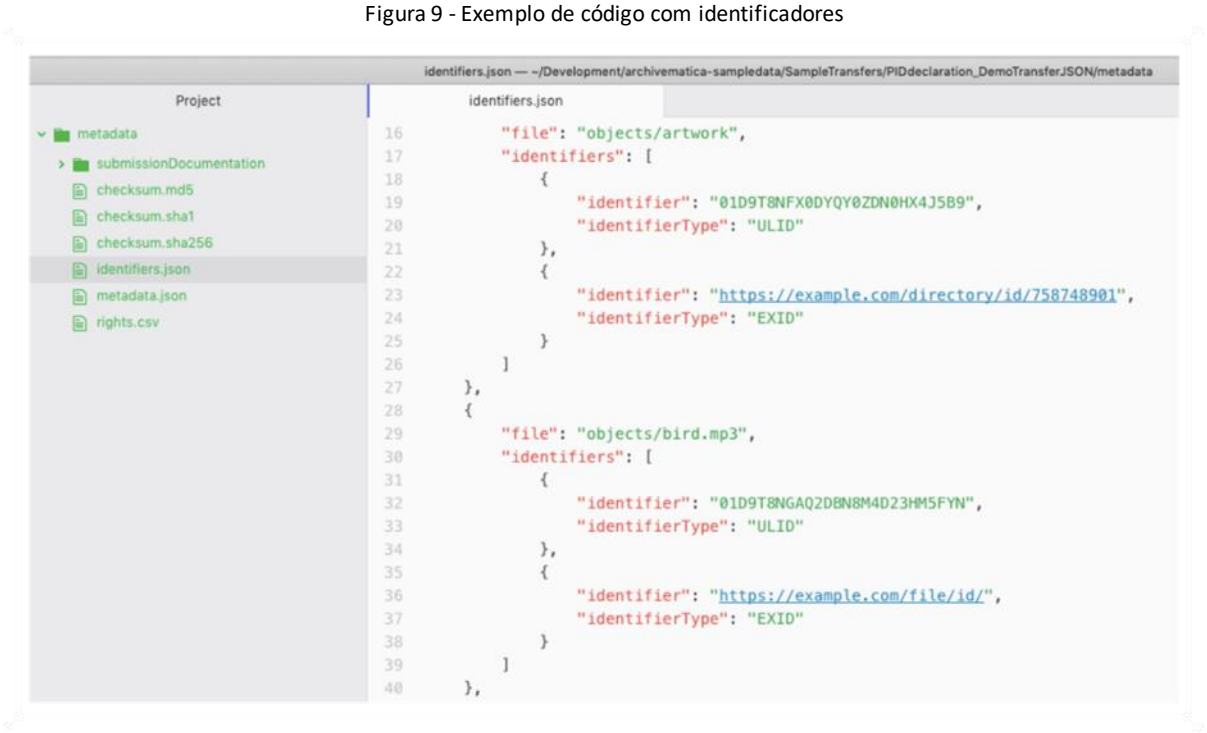
interrompida. A expansão do microsserviço mostrará que o status “Verify metadata directory” em vermelho. Para revisar o erro, basta clicar no ícone da engrenagem do trabalho.

No DJe, a codificação usada para a gerar o hash a ser salvo nos checksums do DJe será md5.

TRANSFERÊNCIAS COM IDENTIFICADORES PERSISTENTES

O Archivematica pode importar identificadores persistentes que foram criados fora do Archivematica, como identificadores DOI ou identificadores criados pelo seu sistema de catalogação. Pode-se vinculá-los aos objetos PREMIS criados pelo Archivematica para seus arquivos, incluindo um arquivo identifiers.json no subdiretório de metadados.

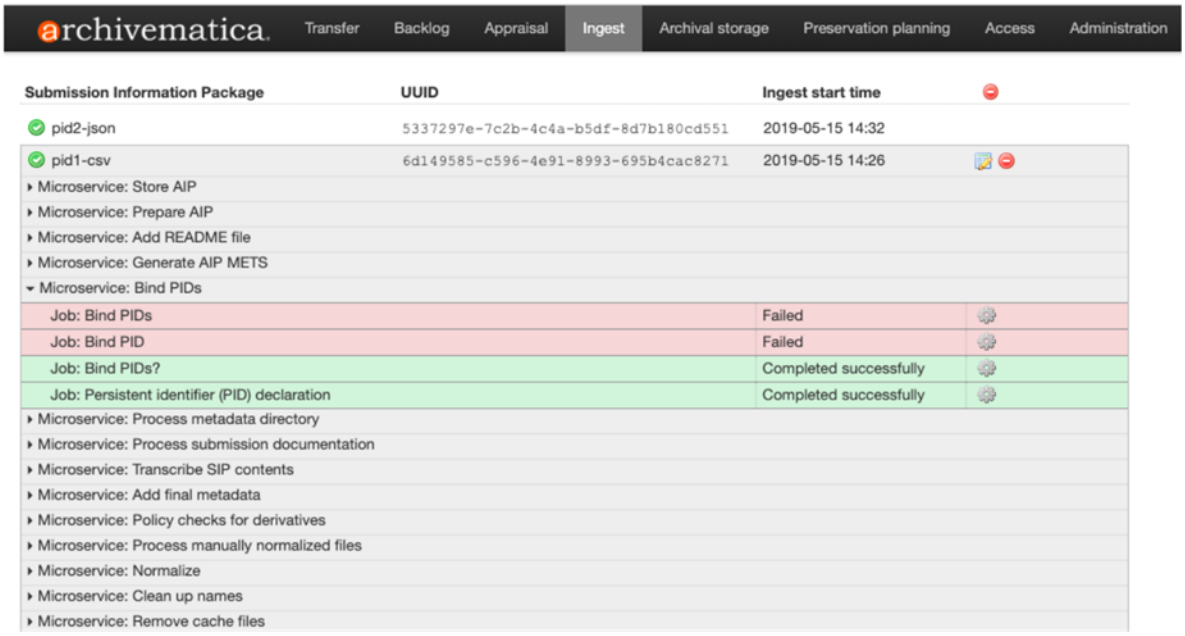
Figura 9 - Exemplo de código com identificadores



Fonte: Archivematica

Esses identificadores persistentes (PID) serão adicionados durante o microsserviço “BIND PIDs”, independentemente do microsserviço estar ativado ou não.

Figura 10 - Identificadores sendo mostrados durante execução do Ingest



Fonte: Elaboração própria

Após a conclusão do Ingest, o PID será vinculado ao premis: objectIdentifier no arquivo AIP METS.

Figura 11 - Exemplo do arquivo METS

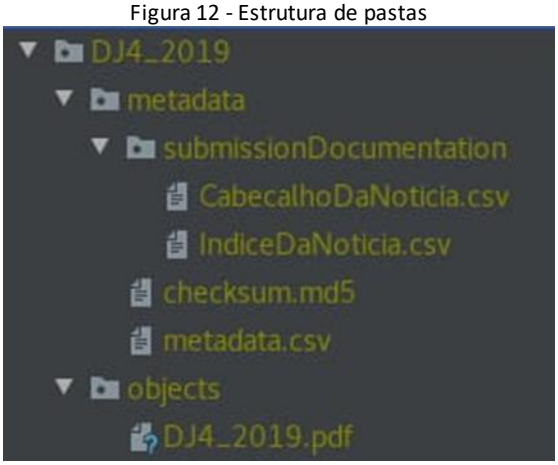
```
<mets:amdSec ID="amdSec_2">
  <mets:techMD ID="techMD_2">
    <mets:mdWrap MDTYPE="PREMIS:OBJECT">
      <mets:xmlData>
        <premis:object xmlns:premis="http://www.loc.gov/premis/v3" xsi:type="premis:file" xsi:schemaLocation="http://www.loc.gov/premis/v3 http://
              www.loc.gov/standards/premis/v3/premis.xsd" version="3.0">
          <premis:objectIdentifier>
            <premis:objectIdentifierType>UUID</premis:objectIdentifierType>
            <premis:objectIdentifierValue>a51f3708-1ae8-4ee0-8b36-cda89466db5b</premis:objectIdentifierValue>
          </premis:objectIdentifier>
          <premis:objectIdentifier>
            <premis:objectIdentifierType>ULID</premis:objectIdentifierType>
            <premis:objectIdentifierValue>01D9T8NH7QF5WW5CJ4KTW15J3C</premis:objectIdentifierValue>
          </premis:objectIdentifier>
          <premis:objectIdentifier>
            <premis:objectIdentifierType>EXID</premis:objectIdentifierType>
            <premis:objectIdentifierValue>https://example.com/file/id/292468806</premis:objectIdentifierValue>
          </premis:objectIdentifier>
          <premis:objectCharacteristics>
            <premis:compositionLevel>0</premis:compositionLevel>
            <premis:fixity>
              <premis:messageDigestAlgorithm>sha256</premis:messageDigestAlgorithm>
              <premis:messageDigest>383d349019ace0e235443c6cb8c5fa3174f00d562281947d36f5fd12aa263687</premis:messageDigest>
            </premis:fixity>
```

Fonte: Archivematica

SOLUÇÃO FINAL DO BARRAMENTO

Para o envio dos metadados, do DJe em formato PDF (assinado), checksums e Submission Documentation, segue abaixo a estrutura de pastas montada pelo barramento:

Figura 12 - Estrutura de pastas



Fonte: Elaboração própria

CÓDIGO DESENVOLVIDO DO BARRAMENTO

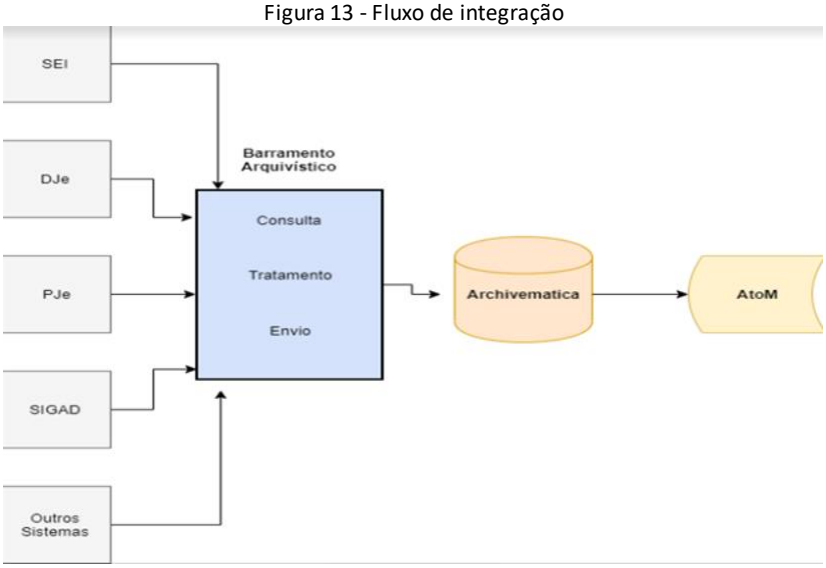
O código gerado durante a pesquisa ainda pode ser encontrado em outros arquivos lógicos de programação, desenvolvidos em linguagem PYTHON. Também é utilizada uma estrutura de funções, conforme se pode observar nos arquivos localizados no barramento, `piloto.py` e `ConsultaBanco.py`, dois arquivos utilizados para ler o XML e o sistema a ser integrado.

Sua composição permite a leitura e composição de saída específica para o Archivematica, copiando e salvando a estrutura entendida pelo sistema de armazenamento e preservação. Os códigos fontes dos dois arquivos mencionados seguem como anexo neste relatório.

APRESENTAÇÕES DE RESULTADOS

Os resultados foram apresentados em reunião e contaram com a equipe do projeto junto ao IBICT e ao TJDFT, demonstrando não somente a coleta de dados do DJe, mas também a execução por meio da utilização de comandos REST ao Archivematica e acesso via AtoM. E também, as pesquisas iniciadas do PJe conforme fluxo a seguir:

Figura 13 - Fluxo de integração



Fonte: Elaboração Própria

Com relação ao PJe, o sistema possui uma complexidade maior que o DJe, advinda, dentre outras coisas, pela mudança no escopo de acesso aos dados e também no formato de retorno obtido. O maior desafio vem do fato de que para se obter os metadados do processo, é necessário mudarmos a forma como o barramento se comunica com o banco de dados, sendo necessária a utilização do sistema de interoperabilidade fornecido pelo CNJ, o MNI.

No barramento adaptado ao PJe, já se é possível efetuar a extração dos dados de forma manual ao se inserir o número do processo. Será necessário, agora, definir a forma de automação da leitura dos processos a serem enviados para guarda permanente, ou, o mecanismo a ser acionado pela equipe de avaliação da necessidade de arquivamento. É preciso, também, implementar um método onde seja possível extrair os anexos do processo que se encontram criptografados em método binário, sua consequente transformação para arquivo acessível e por fim geração do pacote OAIS do Archivematica.

Da mesma forma que ocorre com os anexos, é necessário efetuar os procedimentos citados anteriormente com o próprio processo, pois o mesmo não possui uma forma de arquivo digital em seu banco de dados. Ele é apenas uma forma abstrata de informações provenientes de várias tabelas em seu banco, que são reunidas e exibidas em tela, sem que esta visualização tenha relação com algum pacote digital.

Dentre estes resultados, o treinamento de equipe do TJDFT para o uso e entendimento do Archivematica foi adicionado para o sucesso do projeto, participação e colaboração da equipe do projeto em pesquisar junto a eventos de preservação digital. A capacitação também colaborou para a melhoria constante do desenvolvimento da pesquisa em curso.

Os resultados contemplados no decorrer do projeto, estão sendo organizados e disseminados nas ferramentas

utilizadas até o momento, como o GIT e o REDMINE. Estes conteúdos serão disseminados em plataforma própria ao final da pesquisa.

A ferramenta escolhida para esta disseminação foi um portal formatado como um hotsite, que segue em fase de construção e será disponibilizado antes do final do projeto.

CONCLUSÕES PRELIMINARES

Foi possível implementar diversos avanços tecnológicos no decorrer da meta 3, que culminaram com a implantação do barramento relativo ao DJe e o respectivo processo de implantação do modelo RDC-Arq proposto pelo CONARQ. Os avanços obtidos possuem capacidade de mudar significativamente os processos de guarda permanente de ativos digitais do TJDFT, bem como têm potencial para revolucionar a aplicação do RDC-Arq em instituições públicas e privadas.

A forma como o barramento e sua integração está sendo desenvolvida permite a sua adaptação futura com diversos sistemas. Esta interoperabilidade proposta para o sistema possui embasamento teórico na gestão da informação e nos processos de preservação digital, tão importantes na sociedade atual.

A forma como as equipes do Instituto Brasileiro de Informação em Ciência e Tecnologia e do Tribunal de Justiça do Distrito Federal e Territórios interagiram possibilitou que a maioria dos avanços previstos fossem alcançados nos períodos estipulados, e, quando isso não ocorreu, a motivação foi devido a alguma restrição externa não prevista no planejamento do projeto. Além disso a comunicação entre as equipes foi facilitada pelas reuniões de acompanhamento e pelo estabelecimento de ambientes virtuais de colaboração.

O sucesso da implantação do projeto até o momento reforça a necessidade de se realizar ações de disseminação mais amplas. Além disso, entende-se que a publicação de artigos científicos e livros ainda no escopo do projeto permitirão que o legado obtido seja apropriado por mais instituições. Como estas ações não estavam previstas nas atividades iniciais da meta 4, há a necessidade de ampliação dos prazos do projeto para que elas sejam contempladas.

Por fim, entende-se que todos os resultados obtidos até o momento já correspondem às necessidades do projeto, mas há, ainda, espaço para aprimoramentos. Dessa forma, a expectativa do IBICT é que as temáticas que são objeto do projeto de pesquisa abrirão novos campos de pesquisa, com possibilidade de avanços teóricos e aplicados e com a evolução do processo de gestão de documentos no Tribunal.

ANEXO

Arquivo piloto.py

```
import codecs
import xml.etree.ElementTree as ET
import os
import csv
import json
import requests
import base64
import time
import urllib.request
import schedule
import datetime
import hashlib
import shutil
import ConsultaBanco
# Variaveis globais
colorErro = '\033[91m'
colorPass = '\033[0m'
pastaRaiz = os.path.abspath(__file__)  # Recupera o path do script python
pastaRaiz = (os.path.split(pastaRaiz))[0]  # Recupera o path do projeto
def infoArchivematica(root):
    infoServer = []
    # Recupera informaçoes do servidor do archivematica
    for servidor in root.iter('fileserver'):
        print('Recuperado informaçoes do Archivematica')
        for mapping_tags in servidor:
            infoServer.append(mapping_tags.text)
    return infoServer
def geraMetadatas(root):
    dados = []
    nomeArquivo = []
    nomePasta = []
    PDF = []
    envioSequencial = 2
    submition = []
    nomes_submition = []
    _temp1 = 0
    _temp = []
    # Pega o root do XML e apresenta todas as suas tags
    for principal in root.iter('mapping'):
        print('Gerando metadatas para criação do CSV...')
        # Apresenta todas as tags dentro da MAPPING
        for mapping_tags in principal:
            # Procura a tag METADATAS que possua tabelas associadas
            if mapping_tags.tag == 'metadatas' and mapping_tags.get('table') is
not None:
                assinatura = ConsultaBanco.tabela
                nomeTabela = assinatura[mapping_tags.get('table')]
                if not nomeTabela:
                    print ('Sem dados novos!')
                    return
                # Verifica quais diarios estão assinados
                for i in range(len(assinatura)):
                    # Coleta o numero de dados descritos no envioSequencial
                    if _temp1 == envioSequencial:
```

```
                        break
                    _temp1 = 1 + _temp1
                    metadatas = []   # Inicializa e zera a variavel metadatas
(segunda linha do CSV)
                    identificadorMetadatas = []   # Inicializa e zera a variavel
identificadorMetadatas (primeira linha do CSV)
                    # Recupera todos as linhas do banco que estão com o status
'assinado'
                    for metadatas_tags in mapping_tags:
                        # Pega o texto da tag e divide ele em partes onde possui
espaço, caso possua o espaço, o vetor possuirá tamanho maior que 1
                        variaveis = (metadatas_tags.text).split()
                        # Caso possua duas variaveis no texto, o IF separa cada
uma, caso contrário realiza o ELSE
                        if len(variaveis) > 1:
                            temporaria = ''  # Variável temporaria
                            temporaria1 = []
                            for palavra in variaveis:
                                if palavra[0] != "'":
                                    temporaria                                  +=
str(ConsultaBanco.valor(nomeTabela[i], palavra))
                                else:
                                    temporaria += palavra.replace("'", "")
                                if metadatas_tags.tag == 'pdf':
                                    temporaria1.append(temporaria)
                                    temporaria = ''
                            if metadatas_tags.tag == 'nomeArquivo':
                                nomeArquivo.append(temporaria)
                            elif metadatas_tags.tag == 'nomePasta':
                                nomePasta.append(temporaria)
                            elif metadatas_tags.tag == 'pdf':
                                PDF.append(temporaria1)
                            else:
                                metadatas.append(temporaria)
                                identificadorMetadatas.append(metadatas_tags.tag)
                        else:
                            if metadatas_tags.tag == 'nomeArquivo':
                                nomeArquivo.append(metadatas_tags.text)
                            elif metadatas_tags.tag == 'nomePasta':
                                nomePasta.append(metadatas_tags.text)

                            else:
                              metadatas.append(ConsultaBanco.valor(nomeTabela[i],
                        metadatas_tags.text))
                                identificadorMetadatas.append(metadatas_tags.tag)
                    # Une o identificadorMetadatas (primeira linha do CSV) com os
metadatas (segunda linha do CSV) e armazena em uma nova variavel chamada dados
                    dados.append([identificadorMetadatas] + [metadatas])
            # Procura a tag METADATAS que não possua tabelas associadas
            elif mapping_tags.tag == 'metadatas' and mapping_tags.get('table') is
None:
                temporaria = []
                temporaria1 = []
                for metadatas_tags in mapping_tags:
                    temporaria.append(metadatas_tags.tag)
                    temporaria1.append(metadatas_tags.text)
                for i in range(len(dados)):
                    dados[i][0] = (dados[i][0] + temporaria)
                    dados[i][1] = (dados[i][1] + temporaria1)
            elif mapping_tags.tag == 'submitionDocumentation':
```

```
                for dados1 in range(len(dados)):
                    for submition_tags in mapping_tags:
                        join = ''
                        palavras = (submition_tags.text).split('$')
                        for index, item in enumerate(palavras):
                            if item[0] == '@':
                                for index1, item in enumerate(dados[dados1][0]):
                                    if (item == palavras[index].replace('@', '')):
                                        palavras[index]                          =
dados[dados1][1][index1]
                        consulta = ConsultaBanco.consultas2(join.join(palavras))
                        for row in consulta[0]:
                            _temp.append(dict(zip(consulta[1], row)))
                        if len(nomes_submition) < len(mapping_tags):
                            nomes_submition.append(submition_tags.tag)
                    submition.append(_temp)
                    print(submition)
                    _temp = []
    return dados, nomeArquivo, PDF, submition, nomes_submition
def job(metadatas,infoServer):
    dados = metadatas[0]
    nomeArquivo = metadatas[1]
    PDF = metadatas[2]
    submission = metadatas[3]
    nomes_submission = metadatas[4]
    _temp1 = 0
    _temp = []
    # Escreve o CSV
    for i in range(len(dados)):
        raiz = pastaRaiz + '/documentos/'
        path = raiz + nomeArquivo[i].replace("'", "")  # Caminho onde o arquivo
será criado
        print('Criando estrutura de pastas para envio...')
        if not os.path.isdir(path):
            os.makedirs(path)
        if not os.path.isdir(path + '/metadata'):
            os.makedirs(path + '/metadata')
        if not os.path.isdir(path + '/submissionDocumentation'):
            os.makedirs(path + '/submissionDocumentation')
        # Grava o CSV no diretório designado
        print('Gravando informações no CSV...')
        with open(path + '/metadata/metadata.csv', 'w', newline='') as csvFile:
            writer = csv.writer(csvFile, delimiter=',')
            writer.writerow(dados[i][0])
            writer.writerow(dados[i][1])
        csvFile.close()
        for cont in range(len(nomes_submission)):
            chave = []
            valor = []
            with                                open(path                                +
'/submissionDocumentation/'+nomes_submission[cont]+'.csv',  'w',  newline='')  as
csvFile:
                writer = csv.writer(csvFile, delimiter=',')
                for key,value in submission[i][cont].items():
                    chave.append(key)
                    valor.append(value)
                writer.writerow(chave)
                writer.writerow(valor)
            csvFile.close()
        with codecs.open(path + '/metadata/metadata.csv', 'r', encoding='utf8') as
```

```
transformutf:
            arquivoANSI = transformutf.read()
        with codecs.open(path + '/metadata/metadata.csv', 'w', encoding='utf8') as
transformutf:
            transformutf.write(arquivoANSI)
        transformutf.close()
        try:
            print('\nBaixando Diário ' + str(nomeArquivo[i].replace("'", "")) +
"...")
            urldje = 'https://pesquisadje-api.tjdft.jus.br/v1/diarios/pdf/' +
PDF[i][0] + '/' + PDF[i][1] + '.pdf'
            urllib.request.urlretrieve(urldje, path + '/' +
nomeArquivo[i].replace("'", "") + '.pdf')
        except Exception:
            print (colorErro+"Erro ao baixar PDF não será feito o
arquivamento."+colorPass)
        try:
            with open(path + '/metadata/checksum.sh1', 'w+') as hashmd:
                for root, dirs, files in os.walk(path):
                    for file in files:
                        if (file!='checksum.sh1'):
                            filename = root+'/'+file
                            hasher = hashlib.md5()
                            with open(filename, 'rb') as openfile:
                                content = openfile.read()
                                hasher.update(content)
                                hashDoArquivo = hasher.hexdigest()
                            if (root == path):
                                hashmd.write(hashDoArquivo + '  ' + file + '\n')
                            else:
                                hashmd.write(hashDoArquivo + '  ' +
(root.split(path+'/')[1])+'/'+file + '\n')
                                hashmd.close()
                            openfile.close()
            hashmd.close()
        except Exception:
            print (colorErro+"Erro ao gerar o checksum."+colorPass)
        try:
            print('Enviando para Archivematica')
            url = "http://" + infoServer[0] + ":" + infoServer[1] +
"/api/transfer/start_transfer/?username=" + infoServer[
                2] + "&api_key=" + infoServer[3]
            diretorio = infoServer[4] + ':' + raiz + nomeArquivo[i].replace("'",
"")
            parametros = json.loads(
                '{"name":"' + nomeArquivo[i].replace("'", "") + '","type":"' +
infoServer[5] + '","paths[]":"' + str(
                    base64.b64encode(diretorio.encode("utf-8")), "utf-8") + '"}')
            resposta = (requests.post(url, data=parametros)).json()
        except Exception:
            print (colorErro+"Erro ao iniciar a transferencia para o
Archivematica."+colorPass)

        if resposta['message'] == 'Copy successful.':
            time.sleep(3)
            print('Enviado com sucesso. Realizando a aprovação...')
            urlapprove = "http://" + infoServer[0] + ":" + infoServer[1] +
"/api/transfer/approve/?username=" + infoServer[2] + "&api_key=" + infoServer[3]
            arquivo = resposta['path'].split('/')
            nome = arquivo[len(arquivo) - 2]
```

```
            parametrosapprove = json.loads('{"type":"' + infoServer[5] + '","directory":"' + nome + '"}')
            respostaapprove = (requests.post(urlapprove, data=parametrosapprove)).json()
            print(nome + " - " + respostaapprove['message'])
            print('\n##################Realizando Transfer!############################################################')
            while True:
                urlstatus = "http://" + infoServer[0] + ":" + infoServer[1] + "/api/transfer/status/" + respostaapprove[
                    'uuid'] + "/?username=" + infoServer[2] + "&api_key=" + infoServer[3]
                res = (requests.get(urlstatus)).json()
                try:
                    print("Transfer Status:" + res['status'] + ' as ' + str(datetime.datetime.now()))
                    if (res['status'] == 'COMPLETE'):
                        time.sleep(5)
                        uuidinj = res['sip_uuid']
                        hidetransfer = "http://" + infoServer[0] + ":" + infoServer[1] + "/api/transfer/" + \
                                       respostaapprove['uuid'] + "/delete/?username=" + infoServer[2] + "&api_key=" + \
                                       infoServer[3]
                        (requests.delete(hidetransfer)).json()
                        break
                    elif (res['status'] == 'FAILED' or res['status'] == 'USER_INPUT'):
                        with open('erros.txt', 'a+') as gravarID:
                            gravarID.write("Erro ao efetuar o Transfer: " + str(dados[i][1][0]) + "   " + str(datetime.datetime.now()) + "   " + res['status'] + "\n")
                            gravarID.close()
                        with open('verificadorDeID.txt', 'w+') as gravarID:
                            gravarID.write(str(dados[i][1][1]))
                            gravarID.close()
                        break
                    time.sleep(2)
                except Exception:
                    print (colorErro+"Erro ao verificar o status do TRANSFER."+colorPass)
            print('##################Realizando Ingest!############################################################')
            while True:
                urlstatusinjest = "http://" + infoServer[0] + ":" + infoServer[1] + "/api/ingest/status/" + uuidinj + "/?username=" + infoServer[2] + "&api_key=" + infoServer[3]
                inj = (requests.get(urlstatusinjest)).json()
                print("Ingest Status::" + inj['status'] + ' as '+ str(datetime.datetime.now()))
                if (inj['status'] == 'COMPLETE'):
                    try:
                        hideingest = "http://" + infoServer[0] + ":" + infoServer[1] + "/api/ingest/" + \
                                     uuidinj + "/delete/?username=" + infoServer[2] + "&api_key=" + \
                                     infoServer[3]
                        (requests.delete(hideingest)).json()
                        with open('verificadorDeID.txt', 'w+') as gravarID:
                            print("")
```

```
                            print('Gravando o ID ' + str(dados[i][1][1]) + "\n")
                            gravarID.write(str(dados[i][1][1]))
                            gravarID.close()
                        break
                    except Exception:
                        print    (colorErro+"Erro    ao    gravar    o    ultimo
ID........."+colorPass)
                elif (inj['status'] == 'FAILED' or inj['status'] == 'USER_INPUT'):
                    with open('erros.txt', 'a+') as gravarID:
                        gravarID.write("Erro    ao    gravar    ingest:    "    +
str(dados[i][1][0]) + "   " +  str(datetime.datetime.now()) + "   " + inj['status']
+ "\n")
                        gravarID.close()
                    with open('verificadorDeID.txt', 'w+') as gravarID:
                        gravarID.write(str(dados[i][1][1]))
                        gravarID.close()
                    break
                time.sleep(2)
            else:
                print(resposta['message'])
    # Apaga a pasta documentos ao final do processo
    try:
        shutil.rmtree(pastaRaiz + '/documentos/')
    except OSError as e:
        print(colorErro  +  "Erro:  %s  -  %s.  \n"  %  (e.filename,  e.strerror)  +
colorPass)
    return
def main():
    xml = ConsultaBanco.recuperaXML('dje')
    if xml:
        ConsultaBanco.consultas()
        metadatas = geraMetadatas(xml)
        if metadatas is not None:
            infoServer = infoArchivematica(xml)
            job(metadatas,infoServer)
schedule.every(5).seconds.do(main)
while True:
    schedule.run_pending()
    print("Ultima verificação do sistema:" + str(datetime.datetime.now()))
    time.sleep(1)
```

Arquivo ConsultaBanco.py

```
import pyodbc
import xml.etree.ElementTree as ET
import os
def recuperaXML(nome):
    pastaRaiz = os.path.abspath(__file__)  # Recupera o path do script python
    pastaRaiz = (os.path.split(pastaRaiz))[0]  # Recupera o path do projeto
    root = (ET.parse(pastaRaiz + '/'+nome+'.xml')).getroot()  # Abre o arquivo XML
    return root
def conexaoBanco(root):
    colorErro = '\033[91m'
    colorPass = '\033[0m'
    infoBanco = ''
    # Recupera informações de conexão com o banco de dados
    for database in root.iter('conections'):
```

```
        for mapping_tags in database:
            infoBanco += mapping_tags.tag + "=" + mapping_tags.text + ";"
    # Realiza a conexao com o banco
    try:
        cursor = (pyodbc.connect(infoBanco)).cursor()
        print(colorPass+'Conexão com o banco realizada com sucesso!')
        return cursor
    except pyodbc.Error:
        print(colorErro + 'Erro na conexão com o banco de dados.')
        return
def verificadorDeID():
    colorErro = '\033[91m'
    colorPass = '\033[0m'
    # Abre o arquivo com o ultimo ID utilizado para dar continuidade
    try:
        with open('verificadorDeID.txt', 'r') as numID:
            IDContinua = str(numID.read())
            if (IDContinua == ''):
                print(colorErro + 'Arquivo de verificadorDeID em branco.')
                return 0
            if not IDContinua.isdigit():
                print(colorErro + 'Arquivo de verificadorDeID está com caracter
inválido!')
                return 0
            print(colorPass + 'Leitura do arquivo realizada com sucesso. Continuar
no ID ' + str(int(IDContinua)+1))
            return IDContinua
    except EnvironmentError:
        print(colorErro + 'Arquivo de verificadorDeID não localizado!')
        return 0
# retorna o valor, fornecida o objeto e o nome da coluna
def valor(objeto,coluna):
    exec('global tempValor; tempValor=objeto.'+coluna)
    return str(tempValor)
def consultas():
    IDContinua=verificadorDeID()
    for metadatas in root.iter('metadatas'):
        if metadatas.get('table') is not None:
            cursor.execute('select * from '+metadatas.get('table')+' where id
>'+str(IDContinua)+' and StatusDoDiario="assinado"')
            tabela.update({metadatas.get('table'):cursor.fetchall()})
def consultas2(consulta):
    _temp=[]
    return cursor.fetchall(), columns
tempValor=0
root=recuperaXML('dje')
cursor=conexaoBanco(root)
tabela={}
```