ARQUIVO DISTRITAL DO PORTO

PROJECTO DIGITARQ

MÓDULO CONVERSÃO/DESCRIÇÃO

Relatório final

Porto
2004

Projecto **DigitArq – Produção, conversão e gestão de conteúdos digitais de Arquivo** financiado por:

Programa Operacional da Cultura

FEDER
União Europeia

SUMÁRIO

1. Objectivos

Os objectivos estabelecidos para este subprojecto consistiam em:

1. Converter instrumentos descritivos existentes no Arquivo Distrital do Porto em formatos analógicos e digitais, mas que tinham sido elaborados de forma não normalizada, para um formato normalizado. As normas utilizadas foram: EAD (Encoded Archival Description), ISAD (International Standard Archival Description), ISAAR (cpf) (International Standard Archival Authority Record (corporate, persons, families)) e EAC (Encoded Archival Context). O produto final deste objectivo seria um conjunto textos electrónicos marcados em XML (eXtensible Markup Language) segundo o DTD (Document Type Definition) do EAD

2. Produção de uma ferramenta informática baseada em SQL e XML que permitisse o armazenamento dos textos obtidos, a sua pesquisa expedita e ainda que viabilizasse a produção de descrições arquivísticas a associar aos objectos digitais resultantes, nomeadamente, da digitalização dos documentos custodiados por este Arquivo.

2. Metodologia

A metodologia estabelecida para esta componente do projecto previa a instanciação de diversos passos de planificação de actividades.

2.1 A escolha de metodologia de conversão de ID (instrumentos de descrição) em papel. Havendo como possibilidades a conversão através de OCR e a inserção manual através de trabalho de operadores de todos os textos escritos em papel. A opção seguida foi a de utilizar OCR. As razões para tal residem no facto da equipa de projecto acreditar que este método apresentava benefícios temporais relativamente ao outro e ainda de economia de recursos humanos os quais não podiam de forma alguma ser expandidos.

2.2 A escolha do *software* a ser utilizado para a execução do projecto. Relativamente à utilização da tecnologia XML não havia qualquer dúvida sendo escolhida a ferramenta Xmetal para a produção de DTD e marcação de ID.
No entanto relativamente ao *software* de desenvolvimento da base de dados de suporte ao armazenamento e gestão dos ID convertidos, colocaram-se duas possibilidades: a utilização de uma BD nativa de XML, sendo nesse caso a "TAMINO" a escolha óbvia, ou a utilização de uma plataforma relacional mais convencional. Neste último caso havia ainda as opções da plataforma ORACLE e SQL server. A opção final escolhida foi o SQL server considerando como factores de opção o peso e custo das opções Tamino e ORACLE e ainda a versatilidade e maior simplicidade de utilização do SQL server.

2.3 A escolha dos ID a converter para formatos normalizados. Neste caso foram excluídos todos os ID que se encontravam manuscritos, resumindo-se a conversão aos ID dactilografados ou impressos. Este conjunto constitui a maioria dos ID disponíveis no ADP embora esta escolha tenha sido condicionada pelo facto do OCR não ser eficaz em texto manuscrito. Este é de resto um factor a ponderar para escolher entre o OCR e a digitação manual dos textos integrais.

2.4 O fluxo de trabalho relativo às tarefas encadeadas de digitalização de ID, aplicação de OCR, revisão, anotação e importação

3. Fase 1 do projecto

Este subprojecto dividiu-se em duas grandes fases: uma primeira em que os ID existentes foram convertidos para o formato e tecnologias escolhidas pelo projecto, ou seja, as normas de descrição arquivística ISAD do Conselho Internacional de Arquivos, EAD da SAA e ainda o XML. Na realidade, os produtos finais desta fase foram textos electrónicos estruturados de acordo com o DTD da EAD e anotados com metalinguagem XML.

3.1 Codificação dos elementos EAD

Um primeiro passo consistiu na análise exaustiva e pormenorizada do DTD EAD bem como dos elementos e atributos que o compõem. Os documentos utilizados foram os existentes no sítio *web* da LC (Library of Congress) e ainda do RLG (Research Library Group), *RLG Best Practice Guidelines for EAD*, RLG EAD Advisory Group, Ago 2002. Dada a extensão destes elementos e considerando a necessidade de os inserir nos próprios ID a fim de facilitar o processo subsequente de anotação do texto electrónico através de marcações XML, procedeu-se à atribuição de códigos numéricos estruturados que representassem o elemento e os seus atributos. Este procedimento facilitava a análise e pré-marcação dos ID visto ser apenas necessário escrever um número em vez de uma marca mais extensa. Por ex, em vez de se escrever a etiqueta <materialspec> a marcar um fragmento de texto que contivesse informação correspondente, escrevia-se 442.
Este processo permitiu acelerar o trabalho de pré-marcação e ainda conhecer e familiarizar os elementos do projecto com a norma EAD até aí apenas superficialmente conhecida. A tabela construída encontra-se no anexo A.

3.2 Análise e pré-marcação dos ID

Esta fase do projecto consistiu na observação cuidadosa dos ID seleccionados para conversão — analógicos e digitais — de forma a determinar padrões de distribuição da informação e identificar zonas dentro destes documentos inseríveis nos campos EAD. Procedeu-se ainda à sua pré-marcação colocando em cada zona correspondente a elementos do EAD a identificação numérica correspondente criada na fase anterior (ver ponto 3.1). Esta anotação era colocada a lápis adjacentemente ao bloco de texto analisado. Normalmente os ID repetiam a mesma estrutura informativa de forma que apenas se procedia a marcação nas primeiras páginas de forma a evitar o mais possível criar ruído informativo nos textos analisados. A colocação destes indicadores numéricos auxiliou a fase em que seria necessário marcar as réplicas digitais desses documentos.

3.3 Mapeamento entre os campos do ARQBASE e do EAD

Procedeu-se à análise da estrutura do ARQBASE criando-se correspondência com os campos do EAD. No entanto constatou-se haver muitas discrepâncias e disparidades na forma como os dados tinham sido introduzidos no ARQBASE. Para além do mapeamento directo entre os campos do ARQBASE e do EAD teve que se proceder à identificação da informação inserida nesses campos já que por vezes esta não se adequava ao atributo do campo ou a informação

extravasava o domínio do mesmo. Este facto obrigou à despistagem de todas ocorrências possíveis as quais foram organizadas segundo os critérios de grupo de arquivos e de nível de descrição, uma vez que o preenchimento dos dados no ARQBASE oscilava segundo estas variáveis. Estabeleceram-se ainda, sempre que possível, métricas algorítmicas que viabilizassem o mais possível a automatização da importação.

Ex.:
Sempre que no campo <obs> aparecer subcampo ^e então marcar como <custodhist>

3.4 "Limpeza" da norma.

A norma EAD, ao contrário do que normalmente se espera de um DTD, é flexível permitindo várias opções relativamente aos seus múltiplos elementos. É portanto necessário proceder à sua "limpeza", i. e., a determinação concreta e rigorosa das soluções alternativas a utilizar. Apontamos seguidamente as opções seguidas.

3.4.1 Datas extremas

Um dos primeiros problemas detectados na norma EAD consistia na inexistência de elementos capazes de acolher as datas extremas dos registos convertidos. Por norma, um documento contém duas datas extremas: a inicial e a final. Em alguns casos essas datas poderão coincidir.
A norma EAD apenas contempla a existência de um elemento para representar datas extremas, sendo da responsabilidade do utilizador assegurar a coerência do seu formato. Existem diversas soluções para este problema, no entanto, todas estas recaem sobre dois modelos fundamentais: utilizar uma sintaxe bem definida para as datas que permita distinguir a data inicial da data final, ou utilizar um atributo para indicar a qual das datas o elemento descritivo se refere.

Ex do primeiro caso:
<unitdate>1436/1441</unitdate>.

Ex. do segundo caso:
a representação seria assegurada pelos elementos <unidate datechar='inicial'>1436 </unitdate> e <unidate datechar='final'> 1441 </unitdate>,

onde o atributo datechar indica a que data o elemento unitdate pretende descrever.

Em ambos os casos, a sintaxe escolhida para representar as datas deverá ser bem definida, pois é fundamental que se possam efectuar pesquisas sobre as datas dos registos, tendo obviamente em consideração os intervalos definidos.
Outro problema, que acompanha a codificação de datas, relaciona-se com a representação de informação incompleta, ou seja, como representar as datas para as quais não conhecemos, por exemplo, o dia ou o mês. O formato escolhido deve permitir determinar se uma qualquer data pertence a um intervalo definido por duas datas incompletas.
No que diz respeito às datas, as nossas opções podem ser resumidas da seguinte forma:

- Utilização de atributos para identificar a extremidade da data
  Evita a necessidade de processamento adicional para separar as datas na eventualidade de ser necessário efectuar qualquer tipo de operação com as mesmas.
- As datas são sempre representadas no formato AAAA-MM-DD
  Garante a uniformidade das datas e simplifica todos os cálculos a efectuar com as mesmas.
- Informação incompleta

A informação incompleta é representada por uma cadeia de zeros com o mesmo comprimento do elemento em causa, e. g.., na data 1436-04-00 depreende-se que o dia é desconhecido.

3.4.2 *Numbered* vs *unnumbered components*

No EAD cada nível de descrição pode ser representado de duas formas: utilizando os elementos <c1>, <c2>, ...,<c12>, cujo valor numérico associado representa a profundidade do elemento na árvore, ou utilizando um elemento não numerado representado simplesmente por <c>.

No âmbito do projecto não se viu necessidade de utilizar os elementos numerados, pois a profundidade da hierarquia é automaticamente descrita pela disposição dos diversos elementos no documento XML.

Optamos assim, pela utilização do elemento <c> uma vez que o tratamento numérico da profundidade, para além de supérfluo, implica processamento adicional.

Nos exemplos que se seguem, ambas as representações são perfeitamente equivalentes:

Ex. 1: Anotação usando *numbered components*.

```
<c01>
<c02>
<c03>
...
</c03>
</c02>
<c02>
...
</c02>
</c01>
```

Ex. 2: Anotação usando *unnumbered components*.

```
5
<c>
<c>
<c>
...
</c>
</c>
<c>
...
```

É de notar que o elemento <c> (component) representa um nodo da árvore de descrição documental. Pendurados neste elemento, encontram-se todos os elementos que carregam informação útil produzida pelo arquivista, e. g., o código de referência e o título do elemento descritivo. Um registo cujo código de referência é EMP-BM e cujo título é Banco do Minho podem ser codificados em EAD da seguinte forma:

```
...
<c>
...
<unitid>EMP-BM</unitid>
<unittitle>Banco do Minho<unittile>
...
</c>
...
```

3.4.3 Atributo *otherlevel*

Associado a cada elemento <c> (component) existe um atributo *level* que indica qual o nível de descrição arquivística que o elemento representa. No caso presente foram definidos os níveis de descrição: fundo, subfundo, secção, subsecção, subsubsecção, série, subsérie, unidade de instalação, documento composto e documento simples. O atributo *level* do elemento *component* descrito pela norma EAD/XML define os níveis de descrição: class, collection, fonds, subfonds, series, subseries, file, item, recordgrp , subgrp e otherlevel. Estes níveis não são totalmente

equivalentes aos utilizados no ADP pelo que foi necessário recorrer ao atributo opcional <otherlevel> onde se descreveu os níveis de descrição específicos que se aplicavam.

Ex.: <c level='otherlevel' otherlevel='Fundo'> ...</c>

3.4.4 Elementos mistos

O EAD contempla a utilização de alguns elementos especiais (elementos flutuantes) para anotar secções de texto no interior de outros elementos que se designam por elementos de conteúdo misto. Assim, é possível, por exemplo, na descrição do Âmbito e conteúdo, de um qualquer registo, assinalar que um determinado conjunto de palavras diz respeito ao nome de uma pessoa.

Ex.:
<scopecontent>
O fundador, <person>Manuel Ferreira</person>, constituiu ...
</scopecontent>

A utilidade dos elementos flutuantes neste contexto é amplamente reconhecida, no entanto, a sua implementação do ponto de vista da interface gráfica acarreta alguns problemas, nomeadamente na anotação do texto sem visualizar as etiquetas. Assim, todos os possíveis elementos flutuantes foram substituídos por termos de indexação.

3.4.5 Termos de indexação

Os termos de indexação foram hierarquizados a dois níveis. O primeiro identifica uma série de categorias às quais podemos adicionar termos concretos. Esta opção reduz a ambiguidade inerente a algumas palavras ou expressões.

Por exemplo, o termo "Elias Garcia" pode corresponder ao nome de uma rua ou ao nome de uma escola secundária de Almada.

Ao descrevermos o termo como "Toponímia/Elias Garcia" concluímos imediatamente que se trata do nome de uma rua. Em EAD, o exemplo representar-se-ia na forma ao lado.

```
...
<controlaccess>
<list>
<defitem>
<label>Toponímia</label>
<item>Elias Garcia</item>
</defitem>
</list>
</controlaccess>
...
```

A definição dos termos a utilizar fica a cargo do administrador do sistema. Utilizaram-se no entanto como base as classes de termos preconizadas na norma EAD.

3.5 Importação de ID em papel

O objectivo deste processo, nuclear para o sucesso do projecto, era simplesmente transformar objectos analógicos — impressos e dactilografados — em objectos digitais susceptíveis de serem manipulados informaticamente enquanto texto. Começou-se por digitalizar todos os ID produzindo imagens a preto e branco com resolução média de 200 dpi. (Alguns ID não foram seleccionados por, após um processo de "benchmarking" se verificar que a sua digitalização e conversão por OCR não permitiria obter resultados mínimos vantajosos para a sua conversão em texto editável, à semelhança do que aconteceu com os ID manuscritos). Após todos os ID estarem convertidos em imagens iniciou-se o processo de reconhecimento óptico de caracteres que incluía o seguinte fluxo de trabalho:

1. Leitura da imagem com OCR
2. Correcção do texto obtido
3. Anotação com marcações xml criadas especificamente para o efeito
4. Revisão

Após esta última etapa o resultado final era um texto electrónico revisto e anotado com marcas xml que estava preparado para ser importado para a estrutura de descrição EAD.

Este processo está descrito em pormenor em relatório próprio.

3.6 Importação de ID digitais

Para além dos ID em papel havia vários outros registos em formato electrónico. O mais significativo dos quais era o Arqbase do qual já se falou no ponto 3.3. Havia igualmente informação armazenada em formatos Excel, Access e Word, particularmente os ficheiros provenientes de incorporações efectuadas mas também vários milhares de registos de descrição colocados em bases de dados "artesanais" de Access. Para a migração destes registos foram construídos diversos transformadores específicos para cada formato. O mais complexo destes formatos foi o Arqbase. Neste caso o fluxo de trabalho consistia na importação do formato nativo ISIS para um formato de texto simples através da norma ISO 2709. Seguidamente os textos "brutos" (sem acentuação cedilhas, etc.) assim obtidos eram depurados automaticamente de forma a obter textos "limpos". Após esse trabalho procedeu-se à marcação desses textos segundo o mapeamento e métricas previamente construídas (pontos 3.1, 3.2, 3.3). Foram elaboradas páginas html através da construção de *stylesheets* para permitir a visualização das descrições de forma estruturada segundo a hierarquia de descrição arquivística, e tornar a subsequente despistagem de erros mais fácil. Este processo não foi de todo simples ou totalmente automatizado, implicando em muitos casos a correcção manual de largas porções de informação.
Relativamente aos restantes ID o processo seguiu sensivelmente o mesmo fluxo. Nestes casos o arquivista marcava os textos indicando em linguagem natural os campos que se aplicavam a blocos de informação identificados de forma a dar indicação ao informático qual o posicionamento no DTD que esses mesmos blocos deviam ocupar. Foram construídos transformadores específicos para cada formato electrónico existente.

4. Fase 2 do projecto

Nesta fase procedeu-se ao desenvolvimento de uma ferramenta que permitisse o armazenamento e gestão dos produtos obtidos na fase anterior de forma a viabilizar a pesquisa, recuperação e acesso de forma expedita. Esta aplicação Assentou essencialmente em duas ferramentas; uma base de dados de apoio a descrições arquivísticas e, decorrente dessa, uma aplicação de suporte a incorporações.

4.1 Construção da ferramenta de apoio a incorporações

A construção desta ferramenta não estava planeada inicialmente. O seu desenvolvimento surgiu pelo facto constatado de que as entidades que incorporam documentação faziam-na acompanhar de listagens produzidas em Word e Excel com dados inseridos de forma semi-controlada mas que obrigava sempre à sua posterior digitação para ajustamento com normas de descrição arquivística utilizadas. Havia portanto a necessidade de criar algo que possibilitasse a estas entidades a inserção dos dados normalmente requeridos em incorporações de forma intuitiva e de acordo com as normas de descrição acima referidas e, ainda, que possibilitasse a sua importação automática para o repositório de informação arquivística entretanto constituído. A produção desta ferramenta simples e com reduzidas funcionalidades seria, assim, uma preparação, um "laboratório de ensaio" para a produção da ferramenta mais poderosa para gestão das descrições entretanto convertidas.
Esta aplicação baseia-se numa interface elaborada a partir de VB que corre sobre três estruturas EAD construídas: Uma para os fundos do registo civil, outra para os fundos dos cartórios notariais e uma terceira para os fundos provenientes dos tribunais. Os dados inseridos vão alimentando um ficheiro EAD subjacente que é depois importado para a base de dados central. A ferramenta contém algumas funcionalidades que foram mais profundamente exploradas na aplicação DigitArq como a validação e inferência de dados.

4.2 Construção da base de dados de suporte e armazenamento de ID convertidos

1 Esta base de dados assente em plataforma SQL Server para Microsoft Windows xp. Utiliza ainda tecnologia xml para representar a estrutura arborescente de descrição. Há portanto uma combinação de estrutura relacional (SQL) com estrutura hierárquica (XML)
A escolha de uma base de dados relacional recaiu, em parte, nas condicionantes orçamentais para a aquisição de uma base de dados XML nativa e nas restrições temporais vigentes, que impediam que demasiado tempo fosse dispendido em instalação e configuração. Essa foi uma das razões para a opção pelo SQL em detrimento do ORACLE. Esta última possibilidade foi testada e revelou-se difícil de configurar e manusear e demasiado poderosa e pesada para os objectivos do projecto.
Foi necessário desenvolver um modelo de dados capaz de funcionar sobre uma base de dados relacional que conseguisse reflectir, dentro do possível, a informação contida nos ficheiros EAD/XML resultantes das diversas conversões. A transição de um dos modelos para o outro (XML/Relacional) deveria ser simples e transparente. Foi então desenvolvida uma camada intermédia de *software* para funcionar entre a aplicação de gestão e a base de dados.

2 As descrições arquivísticas baseiam-se em meta-informação sobre os documentos contextualizados numa hierarquia que se inicia no fundo, nível mais lato que corresponde à organização, pessoa ou família produtora, e que desce com mais ou menos nodos intermédios, até ao documento, o nível "atómico" na descrição. Cada registo ou nodo, dessa árvore de descrição encontra-se identificado por um código de referência. Assim, qualquer registo pode ser univocamente identificado pela concatenação das respectivas referências, desde o nível "raiz" até ao documento em causa. Por exemplo, a referência completa BM-L/001/00001, poderá ser interpretada como pertencendo ao fundo BM Banco do Minho), subfundo L (sucursal de Lisboa), série 001 (correspondência recebida) documento 00001 (a referência do documento propriamente dito). Atendendo a esta arquitectura comum a todas as descrições arquivísticas as interfaces gráficas existentes no ADP (à excepção do ARQBASE) de introdução de dados eram baseadas num único formulário onde todos os campos de meta-informação podiam ser introduzidos e onde um dos campos a preencher consistia na referência completa do registo. Isto obrigava o operador a introduzir para cada registo qualquer que fosse o seu nível hierárquico, a referência completa desde a raíz até ao nível onde se encontrava a produzir um registo descritivo. De forma a minimizar a ocorrência de erros decorrentes deste processo de atribuição de referências, construiu-se uma interface gráfica que representa visualmente a árvore de descrição eliminando a necessidade de introdução de longas referências e impedindo o operador de cometer erros aquando da sua introdução. Assim, a interface gráfica está dividida em duas áreas distintas, uma constituída por uma árvore representativa do fundo em que se está a trabalhar e uma outra onde são apresentados os campos que podemos preencher no registo seleccionado, descrevendo o documento ou os níveis do respectivo fundo.

3 Foram identificados dez níveis de descrição distintos: fundo, subfundo, secção, subsecção, subsubsecção, série, subsérie, unidade de instalação, documento composto e documento simples. A interface gráfica assegura a coerência da descrição, impedindo o utilizador de desrespeitar a lógica hierárquica inerente, e. g., a interface não permite a criação de uma secção debaixo de uma subsecção, pois isso violaria a lógica hierárquica subjacente.

4 Além disso, o *software* detecta incoerências e omissões na descrição elaborada através de um processo de validação lançado pelo operador e que vai detectar nas descrições realizados a eventual presença de erros de natureza estrutural e sintáctica. Por exemplo, o não preenchimento de um campo considerado obrigatório de acordo com as normas de descrição utilizadas, ou uma data final inferior a uma data inicial. A realização deste processo assegura a detecção de erros e indica-os ao operador. A sua correcção é apenas realizada de forma manual pelo operador.

5 Foi ainda implementada a funcionalidade de inferência de informação. Esta capacidade consiste num algoritmo que, a pedido do operador e obrigatoriamente depois das descrições terem sido validadas e consequentemente se encontrarem isentas de erros "sintácticos", infere informação contida nos campos <dataincial>, <datafinal>, <unidadeinstalação> e <dimensão>, partindo dos níveis de descrição mais baixos para os mais altos, estabelecendo comparação entre os valores contidos nos campos referidos e colocando-a sucessivamente nos níveis de topo. Deste modo é desnecessário colocar datas ou dimensões aos níveis de descrição de topo visto que estes dados serão "transportados" a partir dos níveis inferiores.

6 O *software* de descrição permite igualmente o trabalho concorrente de vários operadores possibilitando o trabalho simultâneo na BD, inclusive no mesmo fundo. Desta forma, vários operadores podem trabalhar simultaneamente no mesmo acervo documental.

7 Foram desenvolvidos vários relatórios destinados à produção de ID impressos sempre que houver disso necessidade.

Código	*Título*
REL01	Guia de fundos
REL02	Inventário
REL03	Inventário com unidades de instalação
REL04	Catálogo
REL05	Planos de classificação
REL06	Lista de unidades de instalação
REL07	Lista de unidades de instalação para série
REL08	Lista de séries
REL09	Controlo de acessos
REL10	Relatório de localização
REL11	Termos de indexação
REL12	Todos os termos de indexação
REL13	Lista de autores

8 Relativamente ao funcionamento da arquitectura concebida esta é, sucintamente, realizada da seguinte forma:

A camada intermédia (ou *middleware*) é descrita por uma classe abstracta onde são definidos nove métodos fundamentais:

- `Sub Upload()`

Actualiza a informação do nível de armazenamento com a informação residente em memória.

- `Sub Download()`

Actualiza a memória com a informação do nível de armazenamento. Este método permite garantir que a interface apresenta a versão mais recente do registo.

- `Function Children() As LazyNodeCollection`

Retorna uma colecção com os nodos filhos. Se não existirem filhos retorna uma colecção vazia.

- `Sub RemoveChild(ByVal child As LazyNode)`

Remove um filho do nodo actual.

- `Sub AppendChild(ByVal child As LazyNode)`

Adiciona um novo filho ao nodo actual.

- `Function HasChildren() As Boolean`

Retorna verdadeiro se o nodo actual tiver filhos, e falso em caso contrário.

- `Function CreateNode() As LazyNode`

Cria um novo nodo. Este terá que ser adicionado como filho ao nodo pretendido.

- `Function Clone() As LazyNode`

Cria uma cópia do nodo actual.

- `Function Parent() As LazyNode`

Retorna o pai do nodo actual. Caso não exista (nodo actual = raiz) retorna o valor nulo.

9 Para além destes métodos, que cada implementação concreta da classe abstracta obriga, cada nodo carrega consigo um conjunto de propriedades que permitem conservar em memória os valores de cada campo de informação que o nosso sistema é capaz de manipular. A classe abstracta foi baptizada de LazyNode uma vez que a informação apenas é transportada do nível físico para a memória, a pedido. Desta forma, a memória do sistema não é sobrecarregada com informação indesejada.
Foram desenvolvidas duas implementações da classe LazyNode: SQLLazyNode e EADLazyNode. Cada uma das delas limita-se a implementar os nove métodos herdados.

A classe EADLazyNode opera sobre documentos EAD/XML fazendo uso do DOM (W3C Document Object Model) como base de suporte para a manipulação dos ficheiros XML. Assim, todas as operações descritas anteriormente são implementadas com métodos disponibilizados pelo DOM. Existe uma relação directa entre cada propriedade e a sua representação em XML (tabela 1).

Propriedade	*Elemento EAD/*
Otherlevel	@otherlevel
Unitid	did/unitid
CountryCode	did/unitid/@countrycode
RepositoryCode	did/unitid/@repositorycode
...	...

As operações de leitura sobre um documento XML limitam-se a retornar o valor do elemento indicado pelo XPath. Caso o elemento não exista, é retornado o valor nulo. Antes de uma operação de escrita é verificada a existência do XPath correspondente à propriedade que se pretende escrever. Caso não exista, o caminho é criado e o valor do respectivo elemento é actualizado.

10 A implementação do SQLLazyNode permite manipular a informação quando esta se encontra armazenada numa base de dados relacional. Cada nodo da árvore de descrição documental (component <c>) é descrito, no contexto relacional, por um registo que contém uma coluna por cada propriedade definida. Para além das propriedades, foi necessário adicionar alguns campos de controlo: <id>, <Parentid> e <HasChildren>.
O modelo hierárquico inerente à árvore de descrição documental é assegurado por uma relação circular, onde o campo <Parentid> de um registo aponta para o id do registo hierarquicamente superior. Um registo raiz possui um apontador nulo (Parentid = NULL), ou seja, não possui pai. Foi utilizado um campo adicional de controlo designado HasChildren (boleano) para indicar se um determinado nodo da árvore possui ou não filhos. Assim, é possível obter esta informação sem sobrecarregar a base de dados com uma consulta demasiado pesada.

11 Foi ainda integrada na base de dados um conjunto de tabelas destinadas a receber dados relativos aos registos de autoridade (ver anexos). Esta estrutura baseou-se na norma ISAAR (CPF) e no seu DTD equivalente – o EAC (Encoded Archival Context). Os registos de autoridade são inseridos apenas ao nível de fundo e subfundo, visto que incidem sobre as entidades produtoras e não sobre a documentação propriamente dita. A inserção destes dados está reservada apenas aos administradores da base de dados. Salienta-se que este processo não foi totalmente conseguido pelo facto de, na altura do seu desenvolvimento, existir apenas uma versão alfa do DTD EAC pelo que não se dispunha de uma base definitiva de desenvolvimento. A futura integração com o Sistema LEAF (Linking and Exploring Authority Files)[1]foi considerada julgando-se compensador, apesar dos constrangimentos verificados, desenvolver uma estrutura que pelo menos permitisse a produção semi-normalizada de registos de autoridade.

12 Foram implementados formatos de importação e exportação abertos de forma a permitir a inclusão de quase todo o tipo de estrutura de dados e a troca de informação normalizada.
Os formatos implementados foram os seguintes:

[1] Ver http://www.crxnet.com/leaf/

Importação:

EAD	Importa um texto anotado de acordo com o DTD EAD
Fundo	Permite a importação dentro do DigitArq de um fundo para outro fundo. Resolve os casos em que se verifica que um fundo considerado autónomo constitui, afinal, um subfundo.
Texto separado por tabulações	Importa qualquer tipo de texto desde que separado por tabulações.

Exportação:

EAD	Produz ficheiros de acordo com o DTD EAD
EAD CALM Natural Format	Produz ficheiros directamente exportáveis para o CALM .

ANEXOS

Campos da base de dados

Campo da base de dados	XPATH (.../c|archdesc/)	Domínio ou formato
	processinfo/p/date	AAAA-MM-DD
Abstract	did/abstract	String
AccessRestrict	accessrestrict/p	String
Accruals	accruals/p	String
AcqInfo	acqinfo/p	String
AltFormAvail	altformavail/p	String
Appraisal	appraisal/p	String
Arrangement	arrangement/p	String
BiogHist	bioghist/p	String
CountryCode	did/unitid/@countrycode	"pt"
CustodHist	custodhist/p	String
Dimensions	did/physdesc/dimensions	Float x Float x Float
ExtentBook	did/physdesc/extent[@unit='livro']	Integer
ExtentBox	did/physdesc/extent[@unit='caixa']	Integer
ExtentCapilha	did/physdesc/extent[@unit='capilha']	Integer
ExtentCover	did/physdesc/extent[@unit='capa']	Integer
ExtentFolder	did/physdesc/extent[@unit='pasta']	Integer
ExtentMacete	did/physdesc/extent[@unit='maçete']	Integer
ExtentMaco	did/physdesc/extent[@unit='maço']	Integer
ExtentMl	did/physdesc/extent[@unit='ml']	Float
ExtentRoll	did/physdesc/extent[@unit='rolo']	Integer
ExtentOther	did/physdesc/extent[@unit='other']	Integer

ExtentLeaf	did/physdesc/extent[@unit='folha']	Integer
ExtentPage	did/physdesc/extent[@unit='pagina']	Integer
GenreForm	did/physdesc/genreform	[Nenhuma, Duplicado, Extracto]
GeogName	did/physdesc/geogname	String
LangMaterial	did/langmaterial	String
LegalStatus	accessrestrict/legalstatus	String
MaterialsSpec	did/materialspec	String
Note	note[@label='observacoes']/p	String
OriginalNumbering	note[@label='numeracaooriginal']/p	String
OriginalsLoc	originalsloc/p	String
OriginationAuthor	did/origination[@label='autor']	String
OriginationAuthorAddress	did/origination[@label='moradaautor']	String
OriginationDestination	did/origination[@label='destinatario']	String
OriginationDestinationAddress	did/origination[@label='moradadestinatario']	String
OriginationNotary	did/origination[@label='notario']	String
OriginationScrivener	did/origination[@label='escrivao']	String
OtherFindAid	otherfindaid/p	[Nenhum, Inventário, Listagem, Outro]
OtherLevel	@otherlevel	[F, SF, SC, SSC, SSSC, SR, SSR, UI, DC, D]
PhysFacet	did/physdesc/physfacet	String
PhysLoc	did/physloc	String
PhysTech	phystech/p	String
PreferCite	prefercite/p	String
ProcessInfo	processinfo/p/name	String

RelatedMaterial	relatedmaterial/p	String
Repository	did/repository	String
RepositoryCode	did/unitid/@repositorycode	"adprt"
ScopeContent	scopecontent/p	String
SeparatedMaterial	separatedmaterial/p	String
UnitDateFinal	did/unitdate[@datechar='criacaofinal']	AAAA-MM-DD
UnitDateFinalCertainty	did/unitdate[@datechar='criacaofinal']/@certainty	[yes, no]
UnitDateFinalNormal	did/unitdate[@datechar='criacaofinal']/@normal	AAAA-MM-DD
UnitDateInitial	did/unitdate[@datechar='criacaoinicial']	AAAA-MM-DD
UnitDateInitialCertainty	did/unitdate[@datechar='criacaoinicial']/@certainty	[yes, no]
UnitDateInitialNormal	did/unitdate[@datechar='criacaoinicial']/@normal	AAAA-MM-DD
Unitid	did/unitid	String
UnitTitle	did/unittitle	String
UnitTitleType	did/unittitle/@type	[original , atribuido]
UserRestrict	userestrict/p	String
FilePlan	fileplan/p	String

Tabelas da base de dados	***XPATH (.../c|archdesc/)***	***Domínio ou formato***
[Chronlist Table]	bioghist/chronlist	[Data, Event]
[DAOGrp Table]	daogrp/daoloc	[Title, Href]
[ControlAccess Table]	controlaccess/list/item	[Type, Item]
[Bibliography Table]	bibliography/bibref	[BibRef]

Diagrama E/R da base de dados

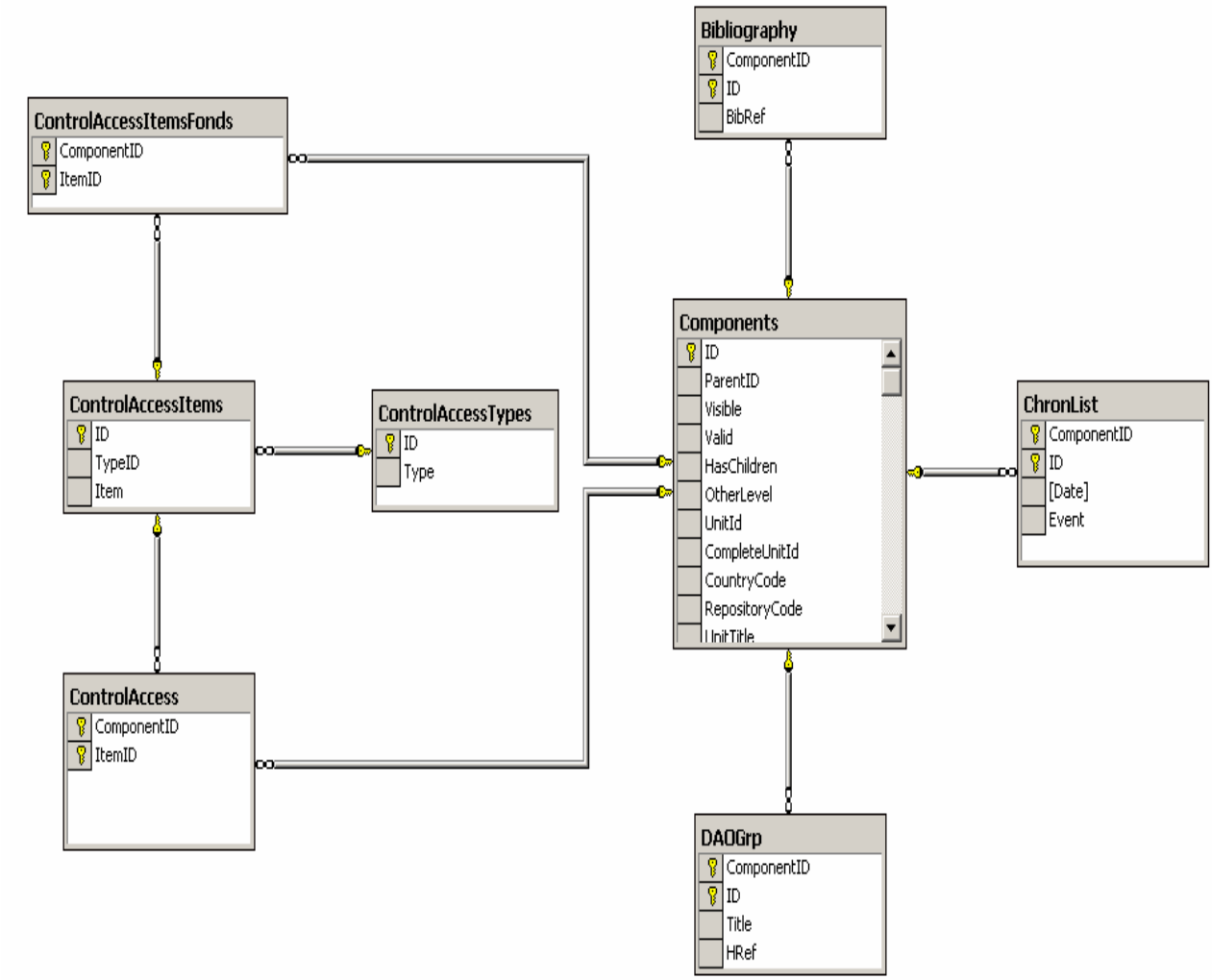

Diagrama E/R da componente EAC