OBJETIVO

Relatar o procedimento de migração dos metadados usados no LILDBI (BVS) para o Repositório Institucional Fiocruz (DSpace).

APLICAÇÃO

Este POP aplica-se a todos que necessitem tal migração

DIVULGAÇÃO

Este POP é divulgado eletronicamente via Repositório Institucional Fiocruz (Arca) ficando disponível para consulta aberta.

EMISSÃO, REVISÃO E APROVAÇÃO

Este POP foi:

- Emitido por: Angelo José Moreira Silva
  Augusto Vinhaes Barboza
  Leonardo Simonini Ferreira
  Vitor Hugo S. Martins
- Revisado por: Paulo Marques
- Aprovado por: Cícera Henrique da Silva

HISTÓRICO

VERSÃO	DATA	PÁGINA	NATUREZA DA MUDANÇA
01	8/ABR/2011	1 a 11	Criação do documento
02	14/ABR/2011	1 a 4	Inclusão dos scripts de migração

PROCEDIMENTO

No sistema LILDBI (BVS/Bireme) foi realizada a extração dos registros de teses defendidas do acervo ENSP e gerado um arquivo com extensão **iso**.

A partir deste arquivo foi utilizado o comando MX para gerar o arquivo dublin_core.i2id com os metadados dos documentos existentes na mesma.

```
mx isso=teses.iso create=teses_ensp now –all
```

O arquivo gerado pelo comando acima foi tratado pelo *shell script* gera_xml_teses.sh constante no Anexo I gerando o arquivo dublin_core.xml

```
./gera_xml_teses.sh  i2id  teses_ensp  teses_ensp.i2id
```

O *script* gera_xml_teses.sh foi rodado novamente com outros parâmetros conforme abaixo, gerando a estrutura de arquivos de publicações de mestres e doutores enviado à equipe do Repositório Institucional Fiocruz (Arca)

```
./gera_xml_teses.sh  teses_ensp.i2id
```

Os arquivos recebidos não foram corretamente apresentados quando abertos em um *browser* e, sendo assim, o dublin_core.xml teve que ser tratado com o *script* abaixo para trocar a codificação de UTF-8 para ISO 8859-1 e para acrescentar o código #38 após o caracter *ampersand*.

```
#!/bin/bash
for i in $(cat arqlista);do
        echo $i
        cd $i
        sed 's/"UTF-8"/"ISO-8859-1"/' < dublin_core.xml  > novo.xml
        sed 's/&/&#38;/' < novo.xml > final.xml
        rm -R dublin_core.xml
        rm -R novo.xml
        mv final.xml dublin_core.xml
        touch contents
        cd ..
done
```

Verificou-se que o arquivo ainda possuía uma série de caracteres de marcação que ao serem importados gerariam um resultado indesejado. Sendo assim, o arquivo foi tratado pelo script abaixo para limpar os caracteres indesejados. Ressalta-se que tais caracteres foram identificados manualmente.

```
#!/bin/bash
for i in $(cat arqlista);do
        echo $i
        cd $i
        sed -e 's/(AU)^ipt//g' -e 's/*^ipt*//g' -e 's/*^btab*//g' \
                -e 's/*^bgraf*//g' -e 's/(AU)//g' \
                -e 's/*.^Ã®pt*//g' -e 's/ ^btab//g' \
                -e 's/ ^bgraf//g' -e 's/ ^ipt//g' \
                -e 's/.^ipt//g' dublin_core.xml > novo.xml
        mv novo.xml dublin_core.xml
        cd ..
done
```

Obs: No script acima foram colocadas quebras de linha, simbolizadas pelo caracter "\" somente para facilitar a leitura. Na reprodução do script é importante que o comando esteja todo em uma única linha.

Após o tratamento do arquivo foram criadas as coleções respectivas no Repositório Institucional Fiocruz (teses e dissertações) na comunidade da ENSP.

De posse da nova estrutura de arquivos de teses e dissertações foram dados dois comandos via linha de comando no Linux para importar estes arquivos.

Ressalta-se que o comando executado (dsrun) pode variar dependendo da coleção e do usuário. Sendo assim, segue abaixo a descrição do mesmo.

```
./dsrun org.dspace.app.itemimport.ItemImport -a -e 1 -c 68 -s [diretorio de importação] -m arquivo.map
```

O parâmetro –a representa o comando add
O parâmetro –e representa "eperson"
O algarismo 1 representa o id interno do usuário que possui direitos de fazer a importação
O parâmetro –c representa "collection" (coleção que se quer importar).
O número 68 representa o id interno da coleção para onde irão os arquivos dentro do banco de dados
O parâmetro –s indica que o texto subseqüente é o diretório de importação.
O parâmetro –m indica que o texto subseqüente é o arquivo de mapeamento gerado.

Após o processo de importação foi realizada uma auditoria manual verificando se havia algum registro duplicado. Nos casos de repetição foi efetuado um processo de remoção manual.

Anexo I

```
#################################################################################################
# Programa: gera_xml_teses.sh
# Autor: Augusto Vinhaes Barboza
# Finalidade: transforma registros da base ISIS em XML - primeiro processamento.
# Execucao: eventual
#################################################################################################

#################################################################################################
# Funcao: finaliza_proc
# Finalidade: grava a citation (fonte), que depende de varios campos; grava o grantor (Fundação
#           Oswaldo Cruz) e finaliza o arquivo xml.
#           Citation: v066 + v062 + v064 + v020 + 'p' + v038
# Paremetros recebidos: -
# Retorno: -
#

function finaliza_proc
{
# Citation

if [ "$v020" != "x" -o "$v038" != "x" -o "$v062" != "x" -o "$v064" != "x" -o "$v066" != "x" ]
then
  CITATION=""
  for Y in "$v038" "${v020}p" "$v064" "$v062" "$v066"
  do
   if [ "$Y" != "x" -o "$Y" != "xp" ]
   then
     CITATION="${Y} ${CITATION}"
   fi
  done
  echo "<dcvalue element=\"identifier\" qualifier=\"citation\">${CITATION}${FECHAXML}" >> $ARQSAI
fi

# Grantor

echo "<dcvalue element=\"degree\" qualifier=\"grantor\">Fundação Oswaldo Cruz${FECHAXML}" >> $ARQSAI

echo $CONST_FIM >> $ARQSAI
}

#################################################################################################
# Funcao: finaliza_ultimo
# Finalidade: processa o ultimo mfn (registro) - necessario devido ao lay-out do arquivo txt
#           gerado pelo comando i2id.
#           Grava a citation (fonte), que depende de varios campos; grava o grantor (Fundação
#           Oswaldo Cruz) e finaliza o arquivo xml.
#           Citation: v066 + v062 + v064 + v020 + 'p' + v038
# Paremetros recebidos: ultimo mfn (registro)
```

```
# Retorno: -
#

function finaliza_ultimo()
{
MFN=$1
CIT=/tmp/cit.txt
rm -f $CIT 2> /dev/null
INI="false"
L_ARQSAI=${DIRSAIDA}/${MFN}/dublin_core.xml
av020=/tmp/av020.txt
av038=/tmp/av038.txt
av062=/tmp/av062.txt
av064=/tmp/av064.txt
av066=/tmp/av066.txt

echo -e "Aguarde, finalizando processamento ... \c"

cat $ARQ | while read REG
do
 if [ "$INI" = "false" ]
 then
   if [ \( "`echo $REG | cut -c1-3`" = "!ID" \) -a \( "`echo $REG | awk '{print $2}'`" = "$MFN" \) ]
   then
           INI="true"; continue
   else
      continue
   fi
 fi

 TAG_ENT="`echo $REG | cut -d'!' -f2`"

 if [ "$TAG_ENT" = "v020" ]
 then
    Z="`echo $REG | cut -d'!' -f3`p "; echo "$Z" > /tmp/av020.txt
 elif [ "$TAG_ENT" = "v038" ]
 then
    Z="`echo $REG | cut -d'!' -f3` ";  echo "$Z" > /tmp/av038.txt
 elif [ "$TAG_ENT" = "v062" ]
 then
    Z="`echo $REG | cut -d'!' -f3` ";  echo "$Z" > /tmp/av062.txt
 elif [ "$TAG_ENT" = "v064" ]
 then
    Z="`echo $REG | cut -d'!' -f3` ";  echo "$Z" > /tmp/av064.txt
 elif [ "$TAG_ENT" = "v066" ]
 then
    Z="`echo $REG | cut -d'!' -f3` ";  echo "$Z" > /tmp/av066.txt
 fi
done
```

```
# Citation

CIT="`head -n1 $av066``head -n1 $av062``head -n1 $av064``head -n1 $av020``head -n1 $av038`"
echo "<dcvalue element=\"identifier\" qualifier=\"citation\">${CIT}${FECHAXML}" >> $L_ARQSAI

# Grantor

echo "<dcvalue element=\"degree\" qualifier=\"grantor\">Fundação Oswaldo Cruz${FECHAXML}" >> $L_ARQSAI

echo $CONST_FIM >> $L_ARQSAI
echo
}

### Inicio  ##############################################################################################

#  Lay-out do arquivo de entrada:
#
#  !ID 000001
#  !v001!BR526.1
#  !v002!26901
#  ...
#  !ID 000002
#  !v001!BR526.1
#  !v002!26902
#  ...

# Tabela de correspondencia de campos do Lildbi / Arca => arquivo tab_teses.txt

TAB=/home/leo/repositorio/teses_ensp/tab_teses.txt
DIRSAIDA="/home/leo/repositorio/teses_ensp/saida"
FECHAXML="</dcvalue>"
CONST_INI="<?xml version=\"1.0\" encoding=\"UTF-8\" standalone=\"no\"?><dublin_core schema=\"dc\">"
CONST_FIM="</dublin_core>"
TAG_ANTERIOR=x
PRIMVEZ="true"
ARQSAIGL=" "
ULTIMO_MFN=/tmp/ultimo_mfn.txt

ARQ=$1
if [ $# -ne 1  ]
then
  echo -e "Sintaxe: gera_xml_teses.sh <arquivo com a saida do comando i2id> !\n"
  exit 99
fi
echo

if [ ! -r $ARQ -o ! -r $TAB ]
then
```

```
  echo -e "Erro: arquivo de entrada ou tabela de equivalencia inexistente !\n"
  exit 99
fi
echo

if [ ! -d $DIRSAIDA ]
then
  echo -e "Erro: diretorio de saida inexistente !\n"
  exit 99
fi

echo

cat $ARQ | while read REG
do
 if [ "`echo $REG | cut -c1-3`" = "!ID" ]
 then
   if [ "$PRIMVEZ" = "true" ]
   then
    PRIMVEZ="false"
   else
    # Grava a citation e finaliza o arquivo xml
    finaliza_proc
   fi

   # Inicia novo arquivo
   #

   MFN=`echo $REG | awk '{print $2'}`
   echo "$MFN" > $ULTIMO_MFN

   if [ ! -d ${DIRSAIDA}/${MFN} ]
   then
     mkdir -m777 ${DIRSAIDA}/${MFN}
     if [ $? -ne 0 ]
     then
       echo -e "Erro na criacao do diretorio ${DIRSAIDA}/${MFN} !\n"
       exit 99
     fi
   fi

   ARQSAI=${DIRSAIDA}/${MFN}/dublin_core.xml
   rm -f $ARQSAI 2> /dev/null
   echo -e "Processando mfn ${MFN} (saida: ${ARQSAI})"

   v020="x"; v038="x"; v062="x"; v064="x"; v066="x"
   echo $CONST_INI >> $ARQSAI
 else
   TAG_ENT="`echo $REG | cut -d'!' -f2`"
```

```
case $TAG_ENT in
  "v016" )
     VAR="`echo $REG | cut -d'!' -f3`"
     TAG_SAI=`grep "^${TAG_ENT} " $TAB | awk '{print $2,$3,$4}'`
          if [ -z "$TAG_SAI" ]
          then
       echo -e "Erro: campo $TAG_ENT inexistente na tabela de equivalencia !"
          else
       echo "${TAG_SAI}${VAR}${FECHAXML}" >> $ARQSAI
     fi
          ;;
  "v018" )
     VAR="`echo $REG | cut -d'!' -f3 | cut -d'^' -f1`"
     TAG_SAI=`grep "^${TAG_ENT} " $TAB | awk '{print $2,$3,$4}'`
          if [ -z "$TAG_SAI" ]
          then
       echo -e "Erro: campo $TAG_ENT inexistente na tabela de equivalencia !"
          else
       echo "${TAG_SAI}${VAR}${FECHAXML}" >> $ARQSAI
     fi
          ;;
  "v019" )
     VAR="`echo $REG | cut -d'!' -f3`"
     TAG_SAI=`grep "^${TAG_ENT} " $TAB | awk '{print $2,$3,$4}'`
          if [ -z "$TAG_SAI" ]
          then
       echo -e "Erro: campo $TAG_ENT inexistente na tabela de equivalencia !"
          else
       echo "${TAG_SAI}${VAR}${FECHAXML}" >> $ARQSAI
     fi
          ;;
  "v020" ) # Nao grava, pois e' parte do citation
     v020="`echo $REG | cut -d'!' -f3`"
          ;;
  "v038" ) # Nao grava, pois e' parte do citation
     v038="`echo $REG | cut -d'!' -f3`"
          ;;
  "v040" )
     W="`echo $REG | cut -d'!' -f3 | tr \[A-Z\] \[a-z\]`"
     if [ "$W" = "en" ]
     then
       VAR="en"
     elif [ "$W" = "pt" ]
     then
       VAR="pt_BR"
     elif [ "$W" = "es" ]
     then
       VAR="es"
```

```
    elif [ "$W" = "fr" ]
    then
      VAR="fr"
    else
            VAR="vazio"
    fi
    TAG_SAI=`grep "^${TAG_ENT} " $TAB | awk '{print $2,$3,$4}'`
          if [ -z "$TAG_SAI" ]
          then
      echo -e "Erro: campo $TAG_ENT inexistente na tabela de equivalencia !"
          else
      echo "${TAG_SAI}${VAR}${FECHAXML}" >> $ARQSAI
    fi
          ;;
"v049" )
    VAR="`echo $REG | cut -d'!' -f3`"
    TAG_SAI=`grep "^${TAG_ENT} " $TAB | awk '{print $2,$3,$4}'`
          if [ -z "$TAG_SAI" ]
          then
      echo -e "Erro: campo $TAG_ENT inexistente na tabela de equivalencia !"
          else
      echo "${TAG_SAI}${VAR}${FECHAXML}" >> $ARQSAI
    fi
          ;;
"v050" )
    VAR="`echo $REG | cut -d'!' -f3`"
    TAG_SAI=`grep "^${TAG_ENT} " $TAB | awk '{print $2,$3,$4}'`
          if [ -z "$TAG_SAI" ]
          then
      echo -e "Erro: campo $TAG_ENT inexistente na tabela de equivalencia !"
          else
      echo "${TAG_SAI}${VAR}${FECHAXML}" >> $ARQSAI
    fi
          ;;
"v051" ) # Aparece duas vezes no arquivo de saida
          # Primeira saida: <dcvalue element="type" qualifier="none">

    W="`echo $REG | cut -d'!' -f3 | tr \[A-Z\] \[a-z\]`"
          if [ "$W" = "mestre" ]
          then
      VAR="Dissertation"
    elif [ "$W" = "doutor" ]
          then
      VAR="Thesis"
    else
            VAR="vazio"
    fi
    TAG_SAI="<dcvalue element=\"type\" qualifier=\"none\">"
    echo "${TAG_SAI}${VAR}${FECHAXML}" >> $ARQSAI
```

```
        # Segunda saida: <dcvalue element="degree" qualifier="level">

    VAR="`echo $REG | cut -d'!' -f3`"
    TAG_SAI="<dcvalue element=\"degree\" qualifier=\"level\">"
    echo "${TAG_SAI}${VAR}${FECHAXML}" >> $ARQSAI
        ;;
"v062" ) # Nao grava, pois e' parte do citation
    v062="`echo $REG | cut -d'!' -f3`"
        ;;
"v064" ) # e' parte do citation, mas tem que gravar na saida
    v064="`echo $REG | cut -d'!' -f3`"
        VAR=$v064
    TAG_SAI=`grep "^${TAG_ENT} " $TAB | awk '{print $2,$3,$4}'`
        if [ -z "$TAG_SAI" ]
        then
      echo -e "Erro: campo $TAG_ENT inexistente na tabela de equivalencia !"
        else
      echo "${TAG_SAI}${VAR}${FECHAXML}" >> $ARQSAI
    fi
        ;;
"v066" ) # e' parte do citation, mas tem que gravar na saida
    v066="`echo $REG | cut -d'!' -f3`"
        VAR=$v066
    TAG_SAI=`grep "^${TAG_ENT} " $TAB | awk '{print $2,$3,$4}'`
        if [ -z "$TAG_SAI" ]
        then
      echo -e "Erro: campo $TAG_ENT inexistente na tabela de equivalencia !"
        else
      echo "${TAG_SAI}${VAR}${FECHAXML}" >> $ARQSAI
    fi
        ;;
"v083" )
    VAR="`echo $REG | cut -d'!' -f3`"
    TAG_SAI=`grep "^${TAG_ENT} " $TAB | awk '{print $2,$3,$4}'`
        if [ -z "$TAG_SAI" ]
        then
      echo -e "Erro: campo $TAG_ENT inexistente na tabela de equivalencia !"
        else
      echo "${TAG_SAI}${VAR}${FECHAXML}" >> $ARQSAI
    fi
        ;;
"v087" )
    if [ "`echo $REG | cut -d'!' -f3 | cut -c1-2`" = "^d" ]
        then
          VAR="`echo $REG | cut -d'!' -f3 | cut -d'^' -f2 | cut -c2-`"
        else
      VAR="`echo $REG | cut -d'!' -f3`"
        fi
```

```
        TAG_SAI=`grep "^${TAG_ENT} " $TAB | awk '{print $2,$3,$4}'`
            if [ -z "$TAG_SAI" ]
            then
          echo -e "Erro: campo $TAG_ENT inexistente na tabela de equivalencia !"
            else
          echo "${TAG_SAI}${VAR}${FECHAXML}" >> $ARQSAI
        fi
            ;;
   esac
 fi
done

# Necessario para gravar a citation do ultimo registro
finaliza_ultimo `head -n1 $ULTIMO_MFN`

exit 0
```