Version number: 0.1

Issue Date: 8 February 2023

# Introduction

## Purpose

The purpose of this procedure is to outline how to use Track Transfer toolset to track objects (normally VEOs) received in transfers of digital records from agencies

## Background

Public Record Office Victoria (PROV) receives transfers of digital records from agencies. These transfers can contain thousands of records received in multiple tranches. Records can be received multiple times (either due to error, or because later instances can replace earlier ones). At the end of the transfer PROV needs to be able to reconcile all of the objects received with those whose custody has been accepted.

This tool set allows the documentation of digital records and generation of reports on the transfer.

## Audience

This procedure has been developed for:

* Transfer archivists in PROV

# Track Transfer toolset

## What is the Track Transfer toolset?

The Track Transfer toolset is a set of tools that allow the tracking, reporting, and reconciliation of digital objects received in a transfer.

## What tools are in Track Transfer toolset?

The Track Transfer toolset contains the following tools:

* New transfer. Run once when a new transfer is started
* New delivery. Run when a new delivery of records has been received to register the objects in the delivery
* Annotate. Run whenever it is necessary to record a comment on objects in the delivery
* Report. Produce a report about the objects, their status, and/or the events that occurred to them.
* Help. A simple summary of available commands and options.

## How are the Track Transfer tools used?

The Track Transfer tools may be run as Java executables or called using an Application Programming Interface (API).

The tools must be run using Java 1.8 or later.

On a computer operating a Windows operating system, the tools can be invoked using the ‘cmd.exe’ program.

A bat file (TT.bat) is provided to simplify the use of the tools.

## Legal

The toolset is licensed under the Creative Commons CC BY 4.0 license. This means that you have a license to do anything that you want with the toolset, provided that you:

* Acknowledge Public Record Office Victoria as the source of the toolset.
* Do not misrepresent the license or your relationship with Public Record Office Victoria.

Specifically, you may:

* Include the code from the toolset in your products, either “as is” or in a modified format.
* Use the code from the toolset as the basis of code in your toolset.

# Track Transfer tool set

## Business rules of a Transfer

To understand the tools and how they are intended to work, it is necessary to understand the conceptual structure of a transfer on which the tools are based.

A Transfer consists of (many, many) Items. Some of these Items are eventually accepted into custody as records, others are discarded. Currently Items are identified by their filename – so two files named ‘VEO-12234.veo.zip’ are considered to be the same Item.

Items are delivered to PROV in Deliveries. A Transfer must have one Delivery, but may have several Deliveries over time. Second and later Deliveries might contain additional or replacement Items. They might also contain duplicate Items.

Because second and later Deliveries may contain Items that are already known about (i.e. replacements or duplicates of files with filenames that have already been seen), we consider Deliveries to contain Instances of Items. In the best of all possible worlds, each Item would be represented by one Instance (received in a Delivery). In the real world, an Item might have multiple Instances. Extremely badly structured Deliveries may even contain duplicate Instances of one Item (i.e. the Delivery contains files with the same file name).

At any given time, each Item has an Active Instance. This Instance is the last received (or encountered in a Delivery). The Active Instance is the Instance being currently worked on. NOTE: This means that if you get the same file in two deliveries, the second file is considered to supersede the first and will be the one being worked on.

An exception to this rule that the final instance is the active instance is if an Instance of an Item is received after the Item has been finalised (i.e. custody has been accepted, or the Item has been judged to be abandoned). In this case, the Instance marked as ‘too late’ and is ignored.

It is possible to annotate Items by either setting a status on the Item, or by associating a comment with the active Instance of the Item. Ultimately it will be possible to annotate an Item by using output from processes (e.g. the status report from the Ingest process, or the VEO validation tools).

## Track transfer commands (general)

There is only one program, the ‘TrackTransfer’ program. All the Track Transfer commands are invoked by specifying them as the first command line argument (e.g. “TT newTransfer…”). The commands are

* NewTransfer: Create a new transfer
* NewDelivery: Create a new delivery within a transfer
* Annotate: Annotate a collection of items within a transfer. The collection is specified by files in a directory
* AnnotateFromFile: Annotate a collection of items within a transfer. The collection is specified by filenames in a file
* Report: Generate a report
* Help: Print a list of the Track Transfer commands.
* DropDatabase: Delete the database. WARNING. This deletes all information about a Transfer, including deliveries, items, instances and event.
* Input: Read a series of Track Transfer commands from a file (intended for testing).

Note that these commands are not prefixed by a ‘-‘ symbol.

## Register a new transfer

This command is used to start the processing of a new transfer. It can only be executed once for each transfer.

The expected model of use is that the transfer archivist will create a transfer directory in which they will put everything related to the transfer. The track transfer database should go in this transfer directory, and there should be only one transfer database in each transfer directory.

The command line arguments are:

* -db <databaseName>: (Mandatory) The name of the database to create. You can use any alpha-numeric string, but avoid punctuation and spaces. You cannot create two databases with the same name in the same directory.
* -desc <description>: (Mandatory) A short description of this transfer (truncated to 100 characters). The string must be enclosed in double quotes.
* -help: (Optional) Print a summary of the command line arguments for newTransfer.

An example of registering a new transfer is:

TT newTransfer -db TR2023-1 -desc “DHS Ward Cards”

Requesting help about the newTransfer command:

TT newTransfer -help

## Register a new delivery

This command is used to process the files received in a new delivery of records received from an agency.

The command line arguments are:

* -db <databaseName>: (Optional) The name of the transfer database to associate this delivery with. The <databaseName> is the same as the <databaseName> you specified in the registerTransfer command. This command line argument is optional. If the current working directory (i.e. the transfer directory) contains just one database, this is assumed to be the desired database.
* -dir <directoryPath>: (Mandatory) The pathname of the directory containing the files received in this delivery.
* -desc <description> (Mandatory) A short description of this delivery (truncated to 100 characters). The string must be enclosed in double quotes.
* -veo: (Optional) Only process VEOs (files that end in ‘.veo’ or ‘.veo.zip’.
* -help: (Optional) Print a summary of the command line arguments for newDelivery.

An example of registering a new delivery is:

TT newDelivery -db TR2023-1 -desc “Test delivery” -dir ./Delivery-20230209

If there is only one transfer database in the current working directory you can omit the -db:

TT newDelivery -desc “Test delivery” -dir ./Delivery-20230209

If you want to only process the VEOs in the directory:

TT newDelivery -desc “Test delivery” -dir ./Delivery-20230209 -veo

Requesting help:

TT newDelivery -help

## Annotate some items

This command annotates all the items found in a specified directory. It will work if the directory contains ‘short-cuts’ to the real files.

If the status is changed to ‘custody-accepted’ the item is locked. No further annotations are allowed, and if a new instance of the item is received in another delivery, the instance is marked as ‘Too Late’.

The command line arguments are:

* -db <databaseName>: (Optional) The name of the transfer database to associate this delivery with. The <databaseName> is the same as the <databaseName> you specified in the registerTransfer command. This command line argument is optional. If the current working directory contains just one database, this is assumed to be the desired database.
* -dir <directoryPath>: (Mandatory) The pathname of the directory containing the files received in this delivery.
* -desc <description> (Conditional) A short description of this delivery (truncated to 100 characters). The string must be enclosed in double quotes. The description becomes an event in the history of the item. At least one of -desc and -status must be present.
* -status <status> (Conditional) Changing the status of the item. The actual status is free text, but the statuses ‘abandoned’ and ‘custody-accepted’ have been predefined. At least one of -desc and -status must be present.
* -veo: (Optional) Only process VEOs (files that end in ‘.veo’ or ‘.veo.zip’.
* -help: (Optional) Print a summary of the command line arguments for annotate.

In the following example, files that are malformed have been collected in the ‘Malformed’ subdirectory:

TT annotate -db TR2023-1 -desc “Not valid PDFs” -dir ./Malformed

Again, the -db can be omitted if there is only one database in the current directory:

TT annotate -desc “Not valid PDFs” -dir ./Malformed

Again, the -db can be omitted if there is only one database in the current directory:

TT annotate -desc “Not valid PDFs” -dir ./Malformed

It is possible to set the status of an item. If the status is set to ‘Custody-Accepted’ no further updates to the Item can be performed:

TT annotate -status Custody-Accepted -dir ./Finished

The other special status is ‘Abandoned’:

TT annotate -status Abandoned -dir ./Junk

It is possible to specify both a status and a description:

TT annotate -status Abandoned -desc “Not permanent records” -dir ./Junk

Requesting help:

TT annotate -help

## Annotate some items based on a file

This command annotates all the items found in a file. The file must be a CSV or TSV file, with one of the columns containing the filename.

If the status is changed to ‘custody-accepted’ the item is locked. No further annotations are allowed, and if a new instance of the item is received in another delivery, the instance is marked as ‘Too Late’.

The command line arguments are:

* -in <filename>: (Mandatory) The file containing the list of items to be annotated. The file is assumed to be a CSV or TSV file (see -csv & -tsv), with one item per line. The initial lines of a file can be skipped (see -skip)
* -itemname <column>: (Mandatory) The column in the file that contains the filenames of the items. The first column is column 0.
* -db <databaseName>: (Optional) The name of the transfer database to associate this delivery with. The <databaseName> is the same as the <databaseName> you specified in the registerTransfer command. This command line argument is optional. If the current working directory contains just one database, this is assumed to be the desired database.
* -skip <count> (Optional). Skip the first <count> lines of the input file (typically header lines). If not present, no lines are skipped.
* -pattern <pattern> (Optional) It is possible to select which lines (Items) in a file to annotate using a pattern. A pattern has the form ‘<columnNo>=<regex>,…’ (see below). If no pattern is specified all lines are selected (subject to the -skip command).
* -tsv (Optional). This file is a TSV file (columns in the file separated by tab characters). If neither -tsv or -csv is specified, the file is assumed to be a TSV file.
* -csv (Optional). This file is a CSV file (columns in the file separated by comma characters). If neither -tsv or -csv is specified, the file is assumed to be a TSV file,
* -desc <description> (Conditional) A short description of this delivery (truncated to 100 characters). The string must be enclosed in double quotes. The description becomes an event in the history of the item. At least one of -desc and -status must be present.
* -status <status> (Conditional) Changing the status of the item. The actual status is free text, but the statuses ‘abandoned’ and ‘custody-accepted’ have been predefined. At least one of -desc and -status must be present.
* -veo: (Optional) Only process VEOs (files that end in ‘.veo’ or ‘.veo.zip’.
* -help: (Optional) Print a summary of the command line arguments for annotate.

In the following example, items that have been abandoned are listed in the CSV file ‘report.csv’. The filenames of the abandoned items are in the first column (column 0) – which may be the only column:

TT annotatefromfile -db TR2023-1 -abandoned -itemcol 0 -in report.csv

Again, the -db can be omitted if there is only one database in the current directory:

TT annotatefromfile -abandoned -itemcol 0 -in report.csv

If the input file is a TSV file:

TT annotatefromfile -abandoned -itemcol 0 -tsv -in report.txt

If the report has a header line, it can be skipped:

TT annotatefromfile -abandoned -itemcol 0 -skip 1 -in report.csv

If you only want Items annotated if the fifth column of the input file contains the value ‘VEO Passed’. Note the used of quotes around the value ‘VEO Passed’:

TT annotatefromfile -itemcol 0 -skip 1 -in report.csv -pattern 4="VEO Passed" -custody-accepted

You can create patterns that select on multiple columns by separating individual patterns by commas. Note that there can be no spaces:

TT annotatefromfile -abandoned -itemcol 0 -skip 1 -in report.csv -pattern 4="VEO Passed",5=”DAS failed”

Note that by selecting using different patterns over several runs of annotateFromFile, you can set annotations on different Items using the same input file. For example, you can select the lines that failed and annotate these as abandoned, and then you can select the lines that succeeded and annotate these as custody-accepted.

Requesting help:

TT annotate -help

## Report on items

This command prints out a report from the database. Four reports are available: complete, custody-accepted, abandoned, and incomplete.

The command line arguments are:

* -db <databaseName>: (Optional) The name of the transfer database to associate this delivery with. The <databaseName> is the same as the <databaseName> you specified in the registerTransfer command. This command line argument is optional. If the current working directory contains just one database, this is assumed to be the desired database.
* -complete: (Optional). A complete report listing each item known. For each item, each instance received is listed and each event that occurred to the item.
* -custody-accepted: (Optional). A report listing every item that has a status of ‘Custody-Accepted’ (case insensitive).
* -abandoned: (Optional). A report listing every item that has a status of ‘Abandoned’ (case insensitive).
* -o <filename>: (Optional) The file to contain the report.
* -help: (Optional) Print a summary of the command line arguments for annotate.

In the following example, files that are malformed have been collected in the ‘Malformed’ subdirectory:

TT report -db TR2023-1 -complete -o ./Full-reconciliation.txt

Again, the -db can be omitted if there is only one database in the current directory:

TT report -complete -o ./Full-reconciliation.txt

Requesting help:

TT report -help

## Drop database

This command deletes a transfer database and all the information contained in it.

The command line arguments are:

* -db <databaseName>: (Mandatory) The name of the transfer database to associate this delivery with. The <databaseName> is the same as the <databaseName> you specified in the registerTransfer command. Unlike the other commands, the -db argument is mandatory in DropDatabase to prevent accidents.
* -help: (Optional) Print a summary of the command line arguments for annotate.

In the following example, files that are malformed have been collected in the ‘Malformed’ subdirectory:

TT dropdatabase -db TR2023-1

## Input

This command a series of track transfer commands from a text file, one per line, and executes them. The command is primarily intended for testing, but could be used for other purposes.

The command line arguments are:

* <file.txt>: (Mandatory) The file containing the Track Transfer commands. Each command is on a separate line, and is not prefixed by ‘TT’ or ‘TrackTransfer’. Blank lines and lines that start with ‘!’ are ignored.

In the following example, files that are malformed have been collected in the ‘Malformed’ subdirectory:

TT input test.txt

End of procedure