Tivoli Application Dependency Discovery Manager v7.3.0.5

Automate downloading and import of DLA books

loadDlaBooks

User's Guide

version 1.0

September 25, 2018

Morten Moeller (moellerm@us.ibm.com)

Preface

Many clients augment the information in the TADDM database by importing Discovery Library Adapter (DLA) books. DLA books represent the ‘latest-and-greatest’ discovery information gathered by specialized programs - Discovery Library Adapters (DLA) - that execute independently of the TADDM environment to gather discovery information and store it in an xml format (the book). IBM provides DLAs that can capture information form specialized systems such as Z/OS environments, IBM Tivoli Network Manager (ITNM), IBM Tivoli monitoring, and others, so that the configuration and relationship information that these environments are aware of, can be captured and stored in a format that TADDM can understand.

One of the challenges for clients that use DLAs is that the DLA books must be made available to the TADDM environment in an automated fashion. Many clients wrap the execution of DLAs in scripts that automatically copy the locally stored DLA books to a shared file system, from which they can be accessed from the TADDM side. However, this option is not available to the Z/OS DLA, which executes in the mainframe environment, and therefore does not have access to a filesystem that can be shared with the distributed environment. To enable storage of the DLA books in a shared file system, the Z/OS DLA provide functions to automatically upload the DLA books to an ftp server. However, there are no facilities in TADDM to automatically download and import DLA books from an ftp server – until now.

The loadDlaBooks utility enables you to automatically

- download DLA books (or other files) from an ftp server
- store them in a file system that is accessible from the TADDM server
- generate delta DLA books
- load the information into the TADDM database
- remove the processed files both on the ftp server and locally

with a single command.

Naturally, the execution of this command can be scheduled – for example by cron – so that the process of importing DLA books becomes fully automated.

Table of Contents

1 TADDM and Discovery Library Books

Almost since its inception, TADDM has supported the loading of externally discovered configuration information from xml files.

For many years, IBM has used the Discovery Library Adapter specifications to exchange IT resource configuration and relationship information between various tools. The DLA architecture this allows for the creation of special programs (Discovery Library Adapters) that extract IT resource configuration and relationship information from a specific source and externalizes this information in a well-defined format in a DLA book. The format of DLA books must adhere to the IdML format, and the DLA architecture defines a strict naming structure for the DLA books to allow consumers to process the DLA books in the sequence in which they were created. THis ensures that the most recent information is consumed last.

Many IBM management platforms, for example IBM Tivoli Monitoring and IBM Tivoli Network Manager, provides DLAs to extract information which can be used to provide resource configuration, relation, and topology information to other management tools – for example Tivoli Business Systems Manager (TBSM), IBM Control Desk (ICD), and Tivoli Application Dependency Discovery Manager (TADDM).

In many instances, the operational monitoring platforms are aware of more details than can be discovered by the TADDM sensors, so it may be preferable to collect such information from the DLA rather than from the TADDM sensor. For example, the ITM DLA provides relationship information regarding which ITM monitoring agents manage which application server instances – which allows you to identify un-managed instances of the application servers (unmanaged DB2 instances, for example.).

In addition, DLAs can be used as the vehicle to populate information into the TADDM database about resources that TADDM cannot discover. For example, TADDM has only limited options discover resources that are hosted on IBM z/Series mainframes. TADDM can discover z/Linux instances, WebSphere Application Server instances hosted on the mainframe, and obtain basic z/Series HW, SysPlex and LPAR resource information leveraging the Enterprise Common Collector. However, TADDM cannot discover CICS regions, DB2 subsystems, IMS address spaces or messaging subsystems sch as MQ and IBM Message Broker. Luckily, IBM offers the z/OS DLA program product offering that is installed on the mainframe systems and collects the detailed information about all the subsystems hosted in a SysPlex. Each time it runs, the z/OS DLA produces one or more DLA books, that are uploaded to a ftp server, from where they can be accessed by the consumers. For obvious reasons, the mainframe does not have access to the distributed file systems, and the distributed tooling does not have access to the mainframe file systems. For that reason, the ftp server is used as the intermediary through which the information is exchanged.

Other DLAs may, or may not, support automated transmission of the DLA books to a central ftp server. However, inf the options is not built into the DLA, it can easily be wrapped into a custom script so that the ftp server hosts the Enterprise DLA Repository.

1.1 Consuming DLA books from TADDM

TADDM includes a utility program – loadidml – that is used to consume DLAs. OIften, this utility is referred to as the *bulk load program*.

For details regarding the use of the loadidml utility, please refer to https://www.ibm.com/support/knowledgecenter/en/SSPLFC_7.3.0/com.ibm.taddm.doc_7.3/UserGuide/t_cmdb_blrunbulkload.html.

1.1.1 Accessing DLA books

The loadidml utility assumes that the DLA books to be consumed is accessible from a file system that is mounted on the system hosting the TADDM server. The implication of this is that somehow the files from the ftp server must be made available to the TADDM server system, either by hosting the ftp server on the TADDM server itself (which is a poor option if the DLA books are consumed by multiple systems) or by remotely mounting the ftp file system on the TADDM system (which may provide challenges based on the technologies (SMB/NFS) challenges, and have security implications). The least intrusive way to make DLA books hosted on an ftp server available to the TADDM server is to download them directly from the ftp server and process them locally.

Automated download of DLA books from an ftp server, and optional removal of the source books after successful download, are some of the core functions provided by the loadDlaBooks utility.

1.1.2 Minimizing load time and resource requirements

Multiple DLA books representing information from the same system, for example a CICS region, a network segment, or an ITM environment, gathered at different times will inherently contain a lot of identical information. To avoid refreshing the same information over and over again, TADDDM offers a utility that compares previous versions of the DLA information with the current DLA book, and generates a delta book. The delta book contains only the updated data (and data required to uniquely identify resources). Typically, only a small fraction is represented in the delta book, and by using this, instead of refreshing all information from the DLA book, the time and resources required to process the information is minimized. To minimize the performance impact for the overall TADDM environment, and in particular the database, it is highly recommended to use delta books whenever possible.

Naturally, the loadDlaBooks utility offers functions to maintain copies of the most current DLA books on the TADDM server, and automated generation and processing of delta books.

1.1.3 Location tagging

The loadidml utility offers a number of options to control the processing – including an option to assign a location to the resources that are processed. This option can be controlled from the loadDlaBooks tool so that a location of your choice can be assigned to the resources that are defined in the set of DLA books that are being processed.

In combination with the selection criteria for the DLA books to process, this allows you to use the loadDlaBooks utility to process only a subset of the available books and assign a custom locationTag to all the resources they contain. Executing multiple invocations of the loadDlaUtility, using different book selection criteria and different locationTags, you have detailed control over the assignment of locations to resources discovered by specific instances of the DLAs, also know as Management Software Systems – or MSS'es.

1.1.4 loadidml arguments

As mentioned, arguments can be parsed to the loadidml utility to control how it processes the DLA books. In addition to the assignment of locationTag, options to using the graph-writhing algorithm (to improve performance) and controls on whether to store copies of the processed DLA boos are available.

These options can also be controlled through the loadDlaBooks utility.

2 Installation

The loadDlaBooks utility has been designed to operate only on a TADDM Storage server to leverage the control information in the TADDM properties files (`collation.properties` and `bulkload.properties`), be able to invoke the loadidml utility, and use the jython executables residing in the TADDM directory structure.

The utility is delivered as a jython script (`loadDlaBooks.jy`) and an accompanying sample properties file (`loadDlaBooks.properties`). By default, the utility will attempt to locate the properties files in the `../etc` directory, relative to the invocation directory.

Note:	In the following, unix syntax is used to reference directory and file names, and in the sample commands. It is assumed that you will be able to translate this to Windows syntax in case you are installing the utility on a Windows platform.

2.1 Prerequisites

Before you install the loadDlaBooks utility, it is recommended that you:

- Tailor the configuration of the bulk load utility to your environment
- Unpack the delta book utility program
- Create the required directory structure for custom scripts and utilities

2.1.1 Bulk load configuration

It is assumed, that in advance of using the loadDlaBooks utility, you have configured the loadidml tool by tailoring the `$COLLATION_HOME/etc/bulkload.properties` file to your environment and verified the settings by executing the idmlload tool successfully.

For details on how to configure the bulk load program, see https://www.ibm.com/support/knowledgecenter/en/SSPLFC_7.3.0/com.ibm.taddm.doc_7.3/UserGuide/c_cmdb_blbulkload.html.

2.1.2 Unpack the delta books utility program

If you plan to use the delta books utility program with the loadDlaBooks utility, the delta books utility must be installed. Archives containing the delta book utility is delivered as part of the TADDM installation, in the `$COLLATION_HOME/tools/deltabooks` directory. Before you can use the delta book utility the archives must be unpacked, and execution rights for the taddm instance user must be configured. For details on how to install and enable the delta book utility program, refer to https://www.ibm.com/support/knowledgecenter/en/SSPLFC_7.3.0/com.ibm.taddm.doc_7.3/UserGuide/c_cmdb_deltabooks.html.

2.1.3 Create custom script directories

The utility must reside in a directory two levels under the $COLLATION_HOME directory and expects to use the following directory structure.

```
$COLLATION_HOME ┐
                └─ <custom> ─┐
                             ├── bin       (hosts the scripts)
                             └── etc       (hosts the properties file)
```

Where `<custom>` denotes your directory for custom scripts and non-standard TADDM utilities. In many implementations the name *custom* is used.

You can create the directory structure by executing the following commands as the TADDM instance user:

```
cd $COLLATIO_HOME
mkdir -p custom/bin
mkdir -p custom/etc
```

2.2 Installation

The installation of the loadDlaBooks utility is completed in a few simple steps performed as the TADDM instance owner:

1. Download the `loadDlaBooks.jy` script accompanying this document to the `$COLLATION_HOME/<custom>/bin` directory.
2. (Unix only) Ensure that the script can be executed by executing the following commands:
   ```
   cd $COLLATION_HOME/custom/bin
   chmod +x loadDlaBooks.jy
   ```

3 (Windows only) If you are installing on a Windows platform, you also need to download the `loadDlaBooks.bat` script accompanying this document to the `$COLLATION_HOME/<custom>/bin` directory.

4 Download the `loadDlaBooks.properties` file accompanying this document to the `$COLLATION_HOME/<custom>/etc` directory.

To verify that the installation is correct, execute the following commands, and verify that the help information for the loadDlaBooks utility is displayed in the termial window:

```
cd $COLLATION_HOME/custom/bin
./loadDlaBooks.jy
```

3 Execution

4 Configuration and use

The operation of the loadDlaBooks utility can be controlled by providing command line arguments, by configuring a properties file that contains custom configurations, or by accepting default values. When the utility evaluates the configuration, command line arguments take precedence over property file settings which take precedence over default values.

The properties and arguments that the utility accepts are grouped into the following logical categories:

- FTP options
- Processing options
- Action options
- Miscellaneous options

The default configuration file that controls the behavior of the loadDlaBooks utility is located in the `../etc` directory relative to the directory hosting the utility script. The name of the control file is `loadDlaBooks.properties`. However, through a command-line argument (-I) you can specify an alternate configuration file to be used for a particular execution of the utility.

In the following each group will be discussed.

4.1 FTP options

The options in this group are used to specify and control the interaction with the ftp server.

Most likely this information will apply to most invocations of the utility, so it is recommended that these options are configured into a properties file.

For reference information regarding the FTP options, please refer to *A.2 FTP options* on page 28.

The valid properties are:

ftpHost	Defines the hostname or IpAddress of the ftp server where the DLA books to process are stored
ftpPort	Port number to be used to access the ftp server. The default value is the unsecure port: 21
ftpUser	Username to be used to authenticate with the ftp server
ftpPassword	Password for the user authenticating with the ftp server
ftpSourceDirectory	The directory, relative to the users home directory on the ftp server in which the DLA books reside.
ftpDebugLevel	Debug level of the ftp interactions performed by the loadDlaBooks utility. Valid options are 0.1. or 2. The default value is 0
ftpRetries	Number of retries to be attempted if the connection to the ftp server cannot be successfully established. Default value is 5
ftpWaitTime	Time to wait between each connection attempt (in seconds). The default value is 30

Most of the options are self-explanatory, but it is worth mentioning that the utility automatically increases the ftpDebugLevel as more and more ftp connection attempts fail. This information will automatically be logged.

4.2 Processing options

The processing options control:

- which configuration file to use
- which action to perform
- which files to download and process
- where files are stored on the local file system
- how the DLA books are processed
- how to cleanup and remove processed files

4.2.1 Specifying a configuration file

Properties that control the loadDlaBooks utility configuration and behavior can be provided as invocation arguments or referenced from a configuration file.

The default configuration file is located in the `../etc/loadDlaBooks.properties` file relative to the location of the loadDlaBooks.jy script.

When automating the DLA processing, you will most likely work with several configuration files, in order to support different use cases.

The utility allows you to specify which control file to use by

To specify a particular control file to be used, apply the `-I` or `–-propertiesFile` command line argument, with a value that references the file to be used:

For example, if you have prepared a configuration file named `CICS.DLA_Control` that resides in the `/tmp` directory, you can instruct the loadDlaBooks utility to use this file by executing the utility using the following commands:

```
cd $COLALTION_HOME/custom/bin
./loadDlaBooks.jy -I /tmp/CICS.DLA_Control
```

When scheduling the invocation by creating a cron entry, use the full path to the utility, as shown below:

```
/opt/IBM/taddm/dist/custom/bin/loadDlaBooks.jy  -I /tmp//tmp/CICS.DLA_Control
```

4.2.2 loadDlaBooks actions

The loadDlaBooks utility offers only two actions:

list lists the files on the ftp server

load downloads and loads DLA book information into the TADDM database

To specify the action to perform use the `-a` or `--action` command line arguments or hardcode your preference in the action property in the control file.

The default action is ‘list’.

4.2.2.1 Testing the behavior

In addition, using the runMode option, you can specify if you want to ‘test’ or ‘execute’ the selected action. When listing files, the runMode setting has no impact, but when you have opted to load DLA information, the runMode controls whether to actually load information into the TADDM database or not. Using the test runMode, the loadDlaBooks utility will perform all normal actions (downloading files, managing versions for delta processing, generating delta books, and removal of work files) **except for**

- loading of book through the loadidml program,

and

- removal of source files from the ftp server.

The default runMode is `test`.

To specify the runMode required to instruct the loadDlaBooks utility to perform the expected actions, provide the value `doit` for the `-r` or `--runMode` command line arguments, or add the line:

```
runMode = doit
```

to your properties file.

4.2.3 Controlling which DLA books to process

The loadDlaBooks utility offers two configuration options to allow you to identify the DLA books hosted on the ftp server to download and process. One option allows you to select only DLA books for which the name of the book file matches a regular expression, and the other allows you to specify the maximum age, in days, of the configuration information in the book files.

The two filtering criteria are mutually inclusive.

Using regular expressions, you can create elaborate filtering to cover most – if not all – of your use cases. In this context, it is important to understand that the DLA architecture prescribes that DLA files adhere to a specific naming standard:

```
<dla name>.<subsystem ID>@<hostname>.<UTC_timestamp>.xml
```

where

<dla name>	represents the provider of the information – typically the logical name of the DLA that has produced the book
<subsystem ID>	represents the identifier from which the information was discovered.
<hostname>	represents the hostname or IpAddress of the system from which the information was gathered
<UTC timestamp>	represents the universal time of the discovery

This naming standard allows you to identify DLA books to be processed by provider (DLA), subsystem, host, and date – or any combination of the components of the name of the DLA book.

On a side note, the combination of the DLA name, the subsystem ID, and the host is commonly referred to as the Management Software System (MSS).

As an example, from the DLA book file named:

```
ZOSDISC310CICS.CICSXX@mainframe.test.com.2018-05-01T14.15.16Z.xml
```

You can tell that the `ZOSDISC301CICS` DLA gathered the information from the `CICSXX` subsystem hosted on `mainframe.test.com` on `May 1st, 2018 at 15 minutes and 16 seconds after 2pm – universal time`.

4.2.3.1 Specifying a naming filter

The targetFileNameRegex property is used to filter in only files that matches the value of the property. As the name suggests, the value must be a matching regular expression.

To specify the criteria used to filter in DLA books to be downloaded and processed, a matching regular expression must be provided as the value for the `targetFileNameFilterRegex` property in a control file, or as the value for the `-F` or `--targetFileNameFilterRegex` invocation arguments.

The matching regular expression provided in the targetFileNameFIlterRegex property is matched against the file names of the DLA books hosted on the ftp server.

By default, if no value is provided for the targetFileNameFilterRegex property, all available files are processed.

Examples of regular expressions that can be used to control which files are processed are:

- `(.*)`
  Identifies all files.

Note:	It is NOT recommended to use this regular expression. If files that do not adhere to the DLA file naming standard exists, the loadDlaUtility is likely to fail if targetFileMaxAge is specified because the timestamp of the file cannot be extracted.

- `(.*\.xml)`
  Identifies all files with the xml file extension
- `(.*\.CICS.*\.xml)`
  all xml files containing the string '.CICS'
  this can for example be used to identify all subsystems for which the subsystem ID starts with the string ‘CICS’ (notice the period preceding the fraction of the name)
- `^((?!CICS).)*.xml$`
  all xml files NOT containing the string 'CICS'
- `(.*@.*\.[0-9]{4}-05-[\d\d].*Z\.xml)`
  all xml files discovered in May (the fifth month)
  notice how the regular expressions last tests for the existence of the @, the 4-digit year, and the 2-digit day, and the Z indicating the universal time zone
- `([a-zA-Z0-9]*)\.([a-zA-Z0-9]*)@([[a-zA-Z0-9*\.]*)\.([0-9]{4}-[0-9]{2}-[0-9]{2})T([0-9]{2}\.[0-9]{2}\.[0-9]{2})Z\.xml`
  generic filter that matches the naming standard

To test and validate your own regular expressions, it is recommended that you use an online regex tester and debugging tool – for example https://regex101.com.

4.2.3.2 Specifying an age filter

The age filter is defined as the value associated with the `targetFileMaxAge` property, or the `-A` or `–-targetFileMaxAge` invocation arguments.

The value is a simple integer representing the maximum age in days for the files to process.

To ensure that only files that have collected information within the last week, use a value of 7.

4.2.4 Download location

When the loadDlaBooks utility have decided which h files to download, they it uses the ftp protocol to get the files into the directory specified by the value associated with the `targetDirectory` property, or the value provided in the `-T` or `-–targetDirectory` invocation arguments.

The value must represent the name of a directory and may be absolute or relative to the invocation directory. No default value is provided.

Assuming, that the same directory will be used for most invocations of the utility, it is recommended that you configure your own default into the configuration file.

A best practice value of `../dlaBooks` will create the `$COLLATION_HOME/custom/dlaBooks` directory, and use this to store all downloaded and generated DLA books.

Notice that all workfiles will automatically be removed after processing to avoid that obsolete files will fill up the file system – unless the `keepWorkFiles` option is enabled,. See *3.2.5 File removal options* on page 18.

4.2.5 Processing options

The loadDlaBooks utility provides options to control the invocation of the loadidml program, as well as options to force the generation of delta books prior to loading the DLA information into the TADDM database. In addition, you can configure the utility to remove the DLA books it has processed from the FTP site, so that the file space usage on the ftp server is kept in check.

4.2.5.1 Bulk load options

In the properties file (or as command line arguments) you can specify processing options that control the invocation of the loadidml program. These options are:

loadidmlOptions	Options to be parsed to the loadidml program at invocation
locationTag	The location to be assigned to the resources loaded from the DLA books

The loadidmlOptions are parsed without validation to the loadidml program. They include arguments the controls the graph loading algoritm, and whether to store copies of the DLA books in the file system configured for the bulk load program. For details see https://www.ibm.com/support/knowledgecenter/en/SSPLFC_7.3.0/com.ibm.taddm.doc_7.3/UserGuide/t_cmdb_blrunbulkload.html.

Because you most likely will use the same set of loadidml options for most infocations of the loadDlaBooks utility, it is recommended that you seacify your preferences in the configuration file by adding a line similar to this:

```
loadidmlOptions = <options>
```

Notice that there is no default value assigned to the loadidmlOptions property.

Note:	You should refrain from specifying the `-f` and `-l` options in the loadidmlOptions property. These arguments are assigned by the loadDlaUtility.

LocationTagging

One of the options that is accepted by the loadidml program is the `-l` argument which is used to specify the location to be assigned to the resources loaded from the DLA books.

While processing different subsets of DLA books, you may desire to assign different locations. To support such operations, the locationTag can be specified in a loadDlaBooks configuration file or provided as a command line argument, so that you can execute the utility multiple times using different configuration files or invocation arguments to support your requirements.

The locationTag is specified as the value applied to the `locationTag` property in the control file, or the `-l` or `--loationTag` command line arguments.

4.2.5.2 Delta book processing

One of the main features of the loadDalBooks utility is that the automated generation of delta books

The following properties govern the delta book processing:

enableDeltaBookProcessing	Flag used to enable the function
deltaBookScriptLocation	Directory to which the delta book utility program has been unpacked

Since both properties will be static in your environment, it is recommended that you specify values for both in your configuration file so that they are used for all invocations of the loadDlaBooks utility.

Delta processing considerations

Using delta books requires additional processing and file space to compare two versions of a DLA book, but the advantage is that the load on the TADDM environment, and especially the database, is minimized. For this reason, it is recommended to use always delta processing.

When delta processing is enabled, a copy of the most recent DLA books that have been downloaded are maintained in the `<targetDirectory>/current` directory.

During delta processing these files will be moved to the `previous` subdirectory, and new DLA books are loaded into the `current` directory. Then, the delta book utility program is invoked to compare the files in the two directories, and generate the delta books. The delta books are stored, along with the log file for the delta book utility program, in the `delta` directory.

Next, the delta books are loaded into the TADDM database (by calling idmlload).

Finally, the DLA books in the previous directory (those that used to be current) are copied to current directory, if the current directory does not contain a mode recent copy. In addition, the files in the current directory are inspected and books that do not represent the most recent version of MSS information (combination of DLA, subsystem, and host) are removed. This ensures that the most recently processed books for each MSS are maintained in the current directory, no matter which filtering criteria was applied to the invocation of the loadDlaBooks utility.

As a result of this mode of operation, you will see the following directories in your file system after having enabled delta book processing.

```
<targetDirectory>─┐
                  ├── current   (most recently processed DLA books)
                  ├── delta     (delta processing log)
                  └── (previous)(temporary work copies of DLA books)
```

If you have not enabled the `keepWorkFiles` option, to avoid filling up your file system, delta books as well as the `previous` directory will be removed.

Enabling delta processing

Before you enable the delta book processing, ensure that the delta book utility program has been unpacked on the TADDM server. See *2.1.2 Unpack the delta books utility program* on page 6.

To enable the delta book processing, simply add the following line to the configuration file:

```
enableDeltaBookProcessing = True
```

Location of the delta book utility program

To specify the location of the delta book utility program, add a line similar to the following to the configuration file:

```
deltaBookScriptLocation = < path to unpacked utility program>
```

The path to the delta book utility program can be absolute or relative to the invocation directory.

For Windows implementations, ensure that you use ‘double slashes’ as directory delimiters.

Assuming, that

- you unpacked the delta books utility program directly in the `$COLLATION_HOME/tools/deltabooks` directory
- the loadDlaBooks utility resides in the `$COLLATION_HOME/custom/bin` directory
- $COLLATION_HOME resolves to `/opt/IBM/taddm/dist`

you can use one of the following ways to specify a value for the deltaBookScriptLocation property:

Unix:

```
deltaBookScriptLocation = ../../tools/deltabooks
deltaBookScriptLocation = $COLLATION_HOME/tools/deltabooks
deltaBookScriptLocation = /opt/IBM/taddm/dist/tools/deltabooks
```

Windows:

```
deltaBookScriptLocation = ..\\..\\tools\\deltabooks
deltaBookScriptLocation = $COLLATION_HOME\\tools\\deltabooks
deltaBookScriptLocation = C:\\IBM\\taddm\\dist\\tools\\deltabooks
```

While you are testing, you can use the -s command line argument to provide a value for the deltaBookScriptLocation property

4.2.6 File removal options

To goes without saying that the exhaustive file manipulation that the loadDlaBooks utility performs may consume a lot of space in the file system, if the files are not managed.

When using delta book processing, it is necessary to maintain local copies of the most current versions of the individual books for each MSS, but besides that, the files that are downloaded should be removed after they have been processed. However, during testing, it may be advantageous to keep local copies of all the processed files, which is why a control property – keepWorkFiles – has been provided to allow you to control whether temporary files are removed automatically.

In addition, unless the DLA books residing on the ftp server are removed by someone, there is a risk that you may download and process the same files over and over again (the DLA process ensures that your database will not be corrupted). Also, if the source files are not removed, the file system on the ftp server may be filled up by obsolete DLA books.

The loadDlaBooks utility offers an option to remove process DLA books after they have been processed. You should use this option with care. If TADDM is the only consumer of the books there are no issued to be concerned about, but if other tools rely on the information in the DLA books hosted on the ftp server, there is a risk that automated removal may impact such solutions.

4.2.6.1 Prevent automated removal of work files

In the event that you want to be able to access the work files after the loadDlaBooks utility has terminated, you should use the `-K` or `--keepWorkFiles` invocation arguments.

The by default work files are always removed. If you want to change this behaviour, add the following line to your control file:

```
keepWorkFiles = True
```

4.2.6.2 Removal of source files

By default, the loadDlaBooks utility does not automatically remove the DLA books that reside on the ftp server. However, in some situations it may be advantageous to automate the removal of these files to prevent processing the same files again and again, and to help optimize the use of the filesystems on the ftp server

To remove source files that have been processed, specify the `-R` or `–removeSourceFiles` invocation arguments. To make this the default behaviour, add the following line to your control file(s):

```
removeSourceFiles = true
```

4.3 Miscellaneous options

The options in this group primarily controls debugging and tracing options, but also includes an option to display all the governing properties in the output – to document the runtime settings.

The following options are offered:

debug	Flag to enable debugging information to be visualized in the console and the log file
help	Displays help information for the tool
quiet	Supress all output to the console
showInvocationOptions	Echo all governing properties in the output
suppressWarnings	Suppress display of warning messages on the console
trace	Same as *debug*

All these properties are treated as Boolean flags, meaning that they do not require values. Bu default they are disabled.

As for all the other governing properties, the values for these properties can be defined in a control file or provided as invocation arguments. If defined in a control file, use a value of `True` or `False` to enable or disable the property. For example, to always show the governing properties in the output, add the following line to the control file:

```
showInvocationOptions = True
```

For shorthand argument mapping see *A.4 Miscellaneous options* on page 30.

5 Execution and automation

The loadDlaBooks utility is designed to be executed similar to most other TADDM utility programs, which implies that it must be run as the TADDM instance account – typically *taddmusr*. However, it can be executed under the control of another user, in which case you must insure that, as a minimum, the $COLLATION_HOME variable is defined, and that the used has read access to the $COLLATION_HOME/bin directory, the directory hosting the loadDlaBooks script, and the location where the dla delta book utility is unpacked, as well as read/write access the $COLLATION_HOME/log, the <custom.>/log directory. and the directory specified by the targetDirectory property.

It should be noted, that the loadDlaBooks utility does not interact directly with the TADDM environment, so TADDM credentials are is not required.

Note: In the following you will find examples on how to provide invocation arguments to control the behavior of the loadDlaBooks utility. For all of these examples it is assumed that values, that are different from the loadDlaUtility default values, have been assigned to the governing properties – except for those related to interaction with the ftp server.

5.1 Manual execution

To execute the loadDlaBooks utility manually, log in as the TADDM instance owner account, move to the directory where the utility resides, and execute it. From a terminal window you would issue the following commands (assuming that the name of the TADDM instance owner account is *taddmusr*, and that the utility resides in `$COLLATION_HOME/custom/bin`):

```
su – taddmusr
cd $COLLATION_HOME/bin
./loadDlaBooks.jy
```

If you follow the example above, the help information will be displayed on the console.

All actions are logged to the standard TADDM log directory in the file named `$COLLATION_HOME/log/loadDlaBooks.log`.

In the following, various sample use cases are provided:

5.1.1 Help and input validation

View help

To view the built-in help.information, invoke the utility without arguments, or use the `-h` or `–help` invocation argument.

```
./loadDlaBooks.jy
```

Viewing input errors

Error messages are displayed at the top of the output, followed by the help information.

For example, if you involve the utility with the `-a get` argument (deliberately requesting a function that is not supported) you will see a message similar to the following at the top of the output:

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Invalid action '-a|--action get' specified.
Choose a value from 'list', 'load'.
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

Vieving governing properties

To see the configuration of the governing properties, you can append the `-O` or `--showInvocationOptions` arguments:

```
./loadDlaBooks.jy -O
```

You will see output similar to the example listed below.

```
INFO:==========================================================================
INFO: Starting loadDlaBooks with options:
INFO: action                      (-a):  help
INFO: debug                       (-d):  False
INFO: deltaBookScriptLocation     (-s):  ../../tools/deltabooks
INFO: enableDeltaBookProcessing   (-D):  True
INFO: ftpDebugLevel               (-f):  0
INFO: ftpHost                     (-H):  192.168.81.131
INFO: ftpPassword                 (-p):  ********
INFO: ftpPort                     (-P):  21
INFO: ftpRetries                  (-n):  5
INFO: ftpSourceDirectory          (-S):  dlaBooks
INFO: ftpUser                     (-u):  taddmusr
INFO: ftpWaitTime                 (-w):  30
INFO: help                        (-h):
INFO: keepWorkFiles               (-K):  True
INFO: loadidmlOptions             (-o):  -g -e
INFO: locationTag                 (-l):  MAINFRAME
INFO: propertiesFile              (-I):  ../etc/loadDlaBooks.properties
INFO: quiet                       (-q):  False
INFO: removeSourceFiles           (-R):  False
INFO: runType                     (-r):  doit
INFO: showInvocationOptions       (-O):  True
INFO: suppressWarnings            (-W):  False
INFO: targetDirectory             (-T):  ../dlaBooks
INFO: targetFileMaxAgeDays        (-A):  None
INFO: targetFilenameFilterRegex   (-F):  (.*.xml)
INFO: trace                               (-t):  False
```

The property values displayed are the ones that are active, based on default values, configuration file properties, and invocation arguments where the invocation arguments takes precedence over configuration file properties, which in turn take precedence over default values.

5.1.2 Listing files on the ftp server

Files hosted on the ftp server can be listed using the `--action list` invocation argument

Notice that because the list action does not change anything in the TADDM database, it does no matter if the runMode is *doit* or *test*.

Listing files on the FTP Server

To list files that are hosted on the ftp server, use the list action.

```
./loadDlaBooks.jy -a list
```

This will apply the filtering that has been configured into the default control file `../etc/loadDlaBooks.properties` in the `targetFileNameFilterRegex` and `targetFileMaxAge` properties.

Note:	Note that the age filtering assumes that the files that are listed adheres to the DLA naming standard, and therefore contains the UTC timestamp as part o the name.

View specific files

To overwrite the default filtering (defined in the configuration file) use the `-A` or `--targetFileMaxAgeDays` properties to change the date filtering, and the `-F` or `--targetFilenameFilterRegex` properties to modify the name-based filtering.

To list the names of the DLA books residing on the ftp server for which the hostname contains *myhost.mynet* and the files are less than 2 days old, use the following invocation:

```
./loadDlaBooks.jy -a list -A 2 -F "(.*myhost\.mynet.*)"
```

Notice how the period (.) in the file name filtering regular expression must be escaped

5.1.3 Loading DLA books

To download DLA books from the ftp server and load the information they contain into the TADDM database, you need to specify the action *load*.

If the runMode property is set to *test*, files will be downloaded, stored in the targetDirectory, and, if deltaProcessing is enabled, deltabooks are generated. In addition, if keepWorkFiles has not been enabled, the work files are removed. In other words, using runMode test, all actions are performed, except for updating the TADDM database and removing files from the ftp server.

Test loading

To run a test run, that allows you to see the files that are downloaded, and the loadidml command that will be executed, invoke the loadDlaBooks utility with the

```
-r test -a load -K -t "../dlaBooks"
```

arguments.

The default value for the runMode property is *test*, but it may have been modified in the configuration file, so you'd better play it safe, and include the -r argument. By default, the values for both the enableDeltaBookProcessing, and the removeSourceFiles properties are *False*. It should be noted, that the loadDlaBooks utility does no offer command line arguments to unset flags, so there is not way to overwrite flags that have been set in a control file.

Using the arguments shown above, files will be stored in the directory referenced by the value of the targetDirectory (-t) property (`../dlaBooks` in this case), and the utility will display the command that would have been executed, if the value of the runMode had been set to *doit*.

Test loading delta books

To enable delta book processing, simply append the `-D` or the `--enableDeltaBookProcessing` argument to the loadDlaBook invocation and specify the location of the delta book utility program using the `-s` argument.

```
./loadDlaBooks.jy
-a load -t "../dlaBooks" -D -s "../../tools/deltatools"
```

Loading DLA information for a location

To instruct the loadDlaUtility to load the resource information in the DLA books, that are downloaded from the ftp server, into the TADDM database and assign a specific locationTag to all resources, ensure that the value for the runMode property is set to doit, and specify the desired locatioTag using the `-l` or `–-locationTag` invocation argument.

```
./loadDlaBooks.jy -a load -r doit -t "../dlaBooks" -l "My location"
```

5.2 Automated execution

The ultimate purpose for the loadDlaBooks utility is to help automate the process of obtaining DLA books, and load the information they contain into the TADDM database.

On a unix platform, this can easily be achieved by enabling the TADDM instance owner to maintain its own cron schedule and schedule the execution of the loadDlaBooks utility in accordance with your requirements. The example below invokes the utility at 00:30 every night, using a custom control file named CICS_DLA.properties, and stores the console output in the standard TADDM log directory:

```
## load DLA books every day at 00:30 every 30 minutes
*30 * * * * /opt/IBM/taddm/dist/custom/bin/loadDlaBooks.jy -I
/opt/IBM/taddm/dist/custom/etc/CICS_DLA.properties
>/opt/IBM/taddm/dist/log/CICS_DLA.out 2>&1
```

Appendix A. Properties reference

The following sections document the properties that can be supplied in the configuration file that governs the operation of the loadDlaBooks utility – or be provided as invocation arguments.

A.1 Action options

The properties listed below control the actions performed by the utility

property	Invocation argument	Value		default value	description
action	-a	list load	lists files hosted on the ftp site downloads files from the ftp site and loads the information into the TADDM database	n/a	Controls which actions are performed by the utility
runType	-r	doit test	executes the actions specified by the action property tests the action specified by the action property without applying updates to the environment	`test`	Allows for testing the invocation to anticipate which actions will be executed
propertiesFile	-I	Path and name – absolute or relative to the invocation directory - of the properties file to be used to govern the invocation		`../etc/loadDlaBooks.properties`	Allows for storing different invocation options so that the utility can easily be automated

Examples:

List files on the ftp site

```
cd $COLLATION_HOME/custom/bin
./loadDlaBooks.jy -a list
```

Load DLA books from the ftp site:

```
cd $COLLATION_HOME/custom/bin
./loadDlaBooks.jy -a load -r doit -I ../etc/myLoad.properties
```

The example above will download all DLA files from the ftp site and load them into the TADDM database, using the governing properties specified in the `$COLLATION_HOME/etc/myLoad.properties` file.

A.2 FTP options

The properties and command line arguments documented below control access to the ftp server, including server location, credentials, file location, and file naming criteria

property	Invocation argument	value	default value	description
ftpHost	-H	Hostname or IPaddress of the ftp server	n/a	Specifies from which ftp server to download DLA books
ftpPort	-P	Port number to be used for the connection to the ftp server	21	Allows for testing the invocation to anticipate which actions will be executed
ftpUser	-u	Username to be used to authenticate with the ftp server	n/a	
ftpPassword	-p	Password for the account used to authenticate with the ftp server	n/a	
ftpDebugLevel	-f	Integer between 0 and 2: 0 No debugging 1 Some debugging 2 Detailed debugging	0	Controls the debug level for the interaction with the ftp server
ftpSourceDirectory	-S	Directory name relative to the home directory of the account used to access the ftp server	n/a	Specifies the location on the ftp server from which to download DLA books
ftpRetries	-r	integer	5	Number of times the utility will attempt to connect to the ftp server
ftpWaitTime	-w	Integer	30	Time between attempts to connect to the ftp server

A.3 Processing options

The options documented below controls the processing of DLA books, and the loading of information into the TADDM database.

Property	Invocation argument	Value	Default value	description
deltaBookScriptLocation	-s	Directory name – absolute or relative to the invocation directory	`n/a`	Identifies the directory to which the utility stores previous versions of DLA files DLA in order to be able to generate delta books
enableDeltaBookProcessing	-D	n/a	`Not enabled`	Flag to enable the generation of delta books prior to loading them into the TADDM database
loadidmlOptions	-o	String containing optional arguments for the loadidml utility	`n/a`	Options to be parsed to the loadidml utility – for example -g to allow for graphic processing Refer to https://www.ibm.com/support/knowledgecenter/en/SSPLFC_7.3.0/com.ibm.taddm.doc_7.3/UserGuide/t_cmdb_blrunbulkload.html for details on the arguments accepted by the loadifml utility
locationTag	-l	string	`n/a`	Specifies the location to be assigned to the resources loaded into the TADDM database through the loadidml utility
keepWorkFiles	-K	n/a	`Not enabled`	Flag to control whether not to remove downloaded files from the local before the utility terminates
removeSourceFiles	-R	n/a	`Not enabled`	Flag to control whether source files on the ftp server are removed before the utility terminates
targetDirectory	-T	Directory name – absolute or relative to the invocation directory	Directory specified by the com.ibm.cdb.bulk.workdir property in the bulkload.properties file	Directory to which the files from the ftp server will be downloaded
targetFilenameFilterRegex	-F	Matching Regular expression	`n/a`	Matching regular expression used to identify files that will be downloaded. For example, `(.*\.xml)` will limit the download to only files from the source directory with an extension of *xml*.

Property	Invocation argument	Value	Default value	description
targetFileMaxAgeDays	-A	integer	n/a	The maximum age, in days, of DLA books to process

A.4 Miscellaneous options

including server location, credentials, file location, and file naming criteria

property	Invocation argument	value	Default value	description
Debug	-d	n/a	Not enabled	Flag to enable debugging
Quiet	-q	n/at	Not enabled	Flag to suppress messages displayed on the console
showInvocationOptions	-O	n/a	Not enabled	Flag to display (and log) the values of properties and invocation arguments governing the actual invocation
suppressWarnings	-W	n/a	Not enabled	Flag enabling the suppression of warning messages
Trace	-t	n/a	Not enabled	Flag enabling detailed debugging

A.5 Sample loadDlaBooks.properties

The following represents to basic content of a configuration file.

```
# [ACTION OPTIONS]
action = list
runType = test

# [FTP OPTIONS]
ftpHost = 192.168.81.131
ftpport = 21
ftpUser= taddm_ftp
ftpPassword=myP@ssw0rd
ftpDebugLevel=0
ftpRetries = 5
ftpWaitTime = 30
ftpSourceDirectory = dlaBooks

# [PROCESSING OPTIONS]
targetFilenameFilterRegex = (.*Z\.xml)
targetdirectory = ../dlaBooks
enableDeltaBookProcessing = True
deltaBookScriptLocation = ../../tools/deltabooks/deltabooks
##   WINDOWS deltaBookScriptLocation = ..\\..\\tools\\deltabooks\\deltabooks
##   UNIX    deltaBookScriptLocation = ../../tools/deltabooks/deltabooks
loadidmlOptions = -g -e
#locationTag = MAINFRAME
keepWorkFiles = False
removeSourceFiles = False

# [MISCELANIOUS OPTIONS]
#debug = false
#quiet = False
#showInvocationOptions = False
#suppressWarnings = False
#trace = False
```

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785 U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement might not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758 U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM‘s application programming interfaces.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information in softcopy form, the photographs and color illustrations might not be displayed.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked