MDCS
Mosaic Dataset Configuration Script
Usage Documentation

Version 18.5
February 9th, 2022

Table of Contents

What is MDCS? **3**

Who should use MDCS? **3**

Why create MDCS? **3**

Input to MDCS **3**

What is a configuration file? **3**

Creating and editing MDCS configuration files **4**

Predefined command codes **4**

User Defined Command codes **7**

How does MDCS work? **8**

MDCS command line arguments **9**

Using Variables **9**

Index Based Multiple Commands **10**

Return Codes **11**

Reporting **11**

Calling MDCS from Batch files **11**

Calling MDCS from Model Builder **11**

Contents of configuration files for MDCS **11**

Example MDCS Project **14**

What is MDCS?

MDCS is an acronym for Mosaic Dataset Configuration Script. It is a python script for creating a mosaic dataset, adding raster's, and setting all required/desired parameters. MDCS can be used to simplify the automation of creating a mosaic dataset, by enabling the workflows to be parameterized and executed in a single step. The automation makes workflows more reproducible and efficient. The parameterization of the workflows also assists in encoding best practices for image management. MDCS is used in a number of the workflows defined in the **Image Management workflows** (link) and **Image Management Guidebook** (link). The parameterization of complete workflows also enables MDCS to be used to document the process of creating mosaic datasets and image services for QA/QC purposes. There are multiple ways in which MDCS can be called such as through GPTools or just using batch files. MDCS can also be called from user interfaces developed to provide simple interfaces to often quite complex workflows.

MDCS is compatible with ArcGIS Pro 1.0 – 2.9.

Who should use MDCS?

MDCS is primarily developed for data managers that wish to automate the process of creating mosaic datasets and services. Developing workflows using MDCS requires knowledge of ArcGIS and a good understanding of how to create and use mosaic datasets. A certain level of expertise at editing XML files is also required. By using MDCS you should be able to define a workflow that can then be easily automated so as to replicate a workflow for different imagery. These workflows can then be run quickly when new imagery is obtained simply by a user (or automated process) running a script or model.

Why create MDCS?

The traditional approach to building a mosaic dataset in ArcGIS requires using multiple GP tools, one at a time, in order to achieve the desired results. This approach works well for creating ad-hoc mosaic datasets. However, this manual approach can be prone to error and is time consuming in cases where multiple similar mosaic datasets need to be created. Often in a manual process some important parameters are not set or documented leading to loss of performance or functionality. MDCS enables workflows for the creation of mosaic datasets to be parameterized for specific types of data and then modified to run on other types (e.g. elevation vs. high resolution satellite data vs. pre-processed orthophotos). MDCS provides a single tool that can replace a large number of GP tools to make the process of creating and configuring mosaic datasets more automated, more repeatable and documented.

Input to MDCS

MDCS takes a configuration file (XML) as input which defines all the necessary processes/GP tools with arguments that are required to create, populate, and configure a mosaic dataset. Each GP tool function predefined in the MDCS library is bound to a unique command code. Using these predefined codes, it is possible to chain together (or omit) required operations to create new mosaic datasets, or to apply changes to an existing mosaic dataset.

What is a configuration file?

The only required input to MDCS at the command line is the XML template file for a given workflow. This configuration file contains all the necessary information to create and populate a mosaic dataset. Information within the configuration file includes the names of GP processes to be executed with their corresponding arguments, mosaic dataset properties, and workspace information. The combination of one or more predefined GP command codes are used to define the entire workflow for creation of one mosaic dataset. In short, each configuration file can be thought

of as a container defining all the required tools for a workflow which can be shared and passed on to anyone else to build a similar mosaic dataset.

Creating and editing MDCS configuration files

The configuration files are XML. In most cases the recommended method for creating such a configuration file is to copy an existing file for a similar workflow and edit it using a standard XML (or text) editor.

In some repeatable workflows, the majority of the parameters remain the same for all mosaic datasets, and only a few of the parameters need to be changed. In such workflows the variables can also be defined as command line parameters, enabling the same configuration file to be used multiple times. The alternative is to create and edit a configuration file for each mosaic dataset that is to be created.

A master configuration file (\scripts\master.xml) is provided. This contains all the available commands and parameters. A valid method of creating a configuration file is therefore to copy the master configuration file, then review and edit all the nodes.

The design of MDCS enables massive scalability to support management and sharing of large imagery collections that may contain imagery from a lot of different sources. One key parameter in the configuration file is *MosaicDatasetType* which has one of three possible values: Source, Derived or Referenced.

The definition and usage of these three different mosaic dataset types is discussed in detail in the Image Management Guidebook in the ArcGIS Help system (link). Essentially a Source mosaic dataset is used for imagery that comes from a single source. In some workflows the imagery from multiple source mosaic datasets are then added to a Derived mosaic dataset. Referenced mosaic dataset are typically used to define separate imagery products. Details are provided in the Guidebook. Sample scripts and data are provided with MDCS, to provide examples proper usage of these types. See the section "**Contents of configuration files for MDCS**" below for further details on the configuration file.

Predefined command codes

The necessary GP tools required to complete a workflow can be chained together by a series of commands. Each GP tool is given a code that is used in the configuration file. Below is a table of the recognized codes and the corresponding GP tool or command along with links to the appropriate section of the help.

Code	Tool	Multi Commands	Newly Added Records Only
ABA	Apply Block Adjustment	Y	N
AF	Add fields	N	N
AI	Adds attribute index on the Mosaic Dataset	N	N
AMD	Analyze Mosaic Dataset	Y	N
AMDS	Alter Mosaic Dataset Schema	Y	N
AMR	Aggregate Multidimensional Raster	N	N
ANCP	Analyze Control Points	Y	N
APCP	Append Control points	Y	N
AR	Add rasters /data to a mosaic dataset	N	N
BB	Build boundary	Y	N

BF	Build footprint	Y	Y
BMDIC	Build Mosaic Dataset Item Cache	Y	N
BMI	Builds multidimensional info in the mosaic dataset	N	N
BO	Build Overviews	Y	N
BP	Build Pyramid	Y	N
BPS	Build Pyramid and Statistic	Y	Y
BS	Build seam lines	Y	N
BSM	Build Stereo Model	N	N
CBA	Compute Block Adjustment	Y	N
CBMD	Color Balance mosaic dataset	Y	N
CC	Compute the min and max cell size ranges	Y	N
CCM	Compute Camera Model	N	N
CCP	Compute Control Points	Y	N
CDA	Compute Dirty Area	Y	N
CF	Compute Fiducials	N	N
CFC	Cache feature class	N	N
CLT	Clear Logs Table	N	N
CM	Create new mosaic dataset	N	N
CP	Compact file Geodatabase	Y	N
CPCSLP	Create Point Cloud Scene Layer Package	N	N
CR	Create new referenced mosaic dataset	N	N
CRA	Copy Raster	N	N
CRTT	Clear Raster Type Table	N	N
CS	Calculate Statistic	Y	N
CSDD	Create Image Service Definition Draft	Y	N
CTP	Compute Tie Points	Y	N
CV	Runs “Calculate values” GP tool to set table values	N	Y
DEL	Delete Data	N	N
DF	Delete Field	N	N
DMD	Delete mosaic dataset	Y	N
DN	Define no data values	Y	Y
DO	Define Overviews	Y	N
EFACP	Export Frame And Camera Parameters	N	N
EMDG	Export mosaic dataset geometry	Y	N
EMDI	Export mosaic dataset items	Y	N
ERF	Edit Raster Function	Y	Y

ETC	Export Tile Cache	N	N
GBAR	Generate Block Adjustment Report	N	N
GEA	Generate Exclude Area	Y	N
GMA	Generate Multidimensional Anomaly	N	N
GPC	Generate Point Cloud	N	N
IFPC	Interpolate From Point Cloud	N	N
IG	Import mosaic dataset geometry	Y	N
IF	Import Fields	N	N
JF	Joins two tables based on a common field	Y	N
MMDI	Merge mosaic dataset items	Y	N
MTC	Manage Tile Cache	N	N
RI	Removes attribute index from existing table	Y	N
RP	Repair Mosaic Dataset Paths	N	N
RR	Register Raster	N	N
RRFMD	Remove raster's from mosaic datasets	Y	N
SE	Sets the geoprocessing environment for raster's	Y	N
SMDI	Split mosaic dataset items	Y	N
SP	Set mosaic dataset properties	N	N
SS	Set statistics for a raster or mosaic dataset	Y	N
STP	Share Package	N	N
STS	Stage a Service Definition	Y	N
SY	Synchronize	N	N
UIO	Update Interior Orientation	N	N
USD	Upload a Service Definition	Y	N

Multi Commands: Defines if the command can be defined multiple times. See Index Based Multiple Commands Section

Newly Added Records Only: Defines if the command works on all records (N) or on only the newly added records (Y). See below.

Set Environment: Call this command before a command that you need to set the environment for. For example. Setting the number of processes for Add Raster's. The SE command needs to be used just before AR in list commands. SE+AR

Publishing Commands: The commands to publish Image services have to be run in the order listed to successfully publish to a server. (CSDD, STS, USD)

CFC Command: This tool is used to create a Cache Metadata Feature Class which can be used with Tiled cache services. The tool uses extents of the raster's and corresponding metadata for the raster's being cached to create a polygon feature class. The polygon feature class and the raster cache can be overlaid in map service to give the user the ability to identify or query the tile cache that is being viewed, thus gaining access to valuable metadata that is usually lost when creating and publishing tile packages. Note for CFC only Mosaic Datasets with mosaic

method NorthWest, ByAttribute, Closest to Viewpoint, or Seamline are supported. The cache must be generated using one of these mosaic methods only.

The commands are executed in the order that they are defined. For each command there is a corresponding node the configuration file that defines the parameters. **Please check 'Contents of configuration files for MDCS' section to understand how to define the parameters for the different commands.**

Multi Commands:

Some commands can be run multiple times in a parameter file. These commands are defined in detail here.

Newly Added Records Only:

When you add data to a mosaic dataset, the following commands (BF, BPS, ERF, DN, CV) will work only on the newly added items. To facilitate this behaviour, when any of these commands appears, MDCS checks to determine if it is preceded by an AR (Add Raster). This is helpful when updating an existing mosaic dataset with new data and running a string of commands.

User Defined Command codes

MDCS allows users to build and use their own commands codes to perform custom functions. This is enabled by defining functions within a python source file called 'MDCS_UC.py'. Once a function is properly defined it can be used similarly to inbuilt MDCS commands. Below is a set of instructions on how this can be achieved.

1. Locate the file called MDCS_UC.py within the scripts folder of your MDCS project.
2. Open this file up in Python editor.
3. Define a command with the class User Code
   A command has to be defined using the following syntax.

```
def sample00(self, data):
        workspace = data['workspace']
        md = data['mosaicdataset']
        log = data['log']
        log.Message('%s\\%s' % (workspace, md), 0)
        return True
```

Where sample00 denotes then custom function name.

The argument type 'data' is a 'dictionary' data type which has the following keys defined.

1. 'workspace'
2. 'mosaicdataset'
3. 'mdcs'
4. 'log'
5. 'base'

The values of the above keys are set automatically by MDCS before each function is called and they are set to *workspace/gdb path*, *mosaic dataset name*, *MDCS XML DOM,* currently *used log object and currently used base object*. For more information regarding this dictionary and how to use it, refer to the sample MDCS_UC.py file.

Defining Parameters for User Defined Commands

The newly defined user defined command may require parameters. These parameters can be defined and read from the config file. To define a new node and items follow these steps.

1. Open up sample parameter XML file.
2. Locate the 'Processes' node within this XML file.
3. Insert your XML structure between < Processes> </ Processes>

Example:

```
<Processes>
        <sample00>
            <mysamplevalue>test</ mysamplevalue >
        </ sample00>
</Processes>
```

The name of the custom node should correspond with the command name.

Calling a User Defined Command with MDCS

A User defined command can be called from MDCS just like any other inbuilt command using either the –c option in the command line or defined within a parameter file.
For example:

python.exe mdcs.py -c: **sample00**

or:

python.exe mdcs.py -c:CM+AR+ **sample00**+BF

How does MDCS work?

MDCS reads in the configuration file specified at the command line input and, based on the GP command codes defined in the configuration file, goes through each one of them in the order they are defined. The order is important, because in some cases changing the order will change the resulting output, including a possible failure of one or more processes (e.g. attempting to *AddRasters* before execution of *CreateMosaic*).

GP codes defined within the configuration file can also be overridden by specifying the codes at the command prompt as input to MDCS. Upon execution, if any codes are defined at the command prompt, all codes within the configuration file will be ignored. This enables you to run just parts of the complete script to verify the intermediate outputs.

MDCS processes commands sequentially by first verifying each command to be valid prior to executing that command. Corresponding information/arguments for the matching GP operation is read from the configuration file to run the desired GP tool. Any errors/warnings, if found, while each process is being run are recorded in the log-file in the \logs\ directory.

MDCS command line arguments

MDCS takes in few arguments at the command line to work with as shown below:

MDCS.py <configuration file> <options>

The available options are

-c: command code(s)

These are a list of override command codes. If used, then all command codes in the input configuration.xml file are ignored. These can be used to call only specific parts of a workflow.

-m: Mosaic dataset path including GDB and MD name [e.g. c:\WorldElevation.gdb\Portland].

When specified overwrites the <WorkspacePath>, <Geodatabase> and <Name> values in the configuration file

-s: Source data paths.

When specified overrides the <Sources> <data_path> value in the configuration file .The node <Sources> can take multiple <data_path> child nodes separated by a ";"

-l: Log file to write to disk [path+file name].

Defines the full name of the log file to create. This enables calling applications to define the name of a log file to be checked after the process is run.

-p: <value>$<variable>

Defines a variable to be used in place of the default in the configuration file.

-artdem:

Defines the DEM to be used in an ArcGIS Raster type file. Use this option only if you would like to overwrite the DEM path defined in the ART file. This option only works for ART files that use the DEM option.
*Note. Instead of using this option a DEM can be alternatively passed in the <auxillary input> section in the "Add Rasters to Mosaic Dataset" Tool. Refer to the master.xml under the scripts folder for more information.

Using Variables

In workflows where only a few parameters change from one mosaic dataset to the next, MDCS supports the use of variables in the configuration file. Variables are used as place holders within the configuration file to substitute actual values that can be specified using the command line when invoking MDCS. Variables can be defined for any node or attribute within the configuration file using the following syntax:

<defaulValue>;$<variableName>$

Where

<defaultValue> can be any string, number or any Enumerated value.

<variableName> can be any unique identifier.

e.g.

```
<sample00>
    <variable>10;$var$</variable>
</sample00>
```

The default value is a fail-safe to be applied in case MDCS fails to get the correct variable/value combination.

Look up the '–p' option here to know more about passing variable values to MDCS. To pass multiple variable values repeat the '-p' option for each variable/value combination.

Index Based Multiple Commands

MDCS can accept multiple instances of the same command. This can be required in some cases. Commands can be defined multiple times using different parameters using an indexing system added to the command. The indexing system is zero-based, however the zero is not specified in the command.

An example of this is to run a Calculate Values twice. The typical command sequence would be as follows: CM+AR+**CV**+BF+**CV1**. CV and CV1 will have unique nodes within the configuration file and would typically look like this:

```
<CalculateValues>
    <CalculateValue>
      <query>Category = 1</query>
      <fieldName>MaxPS</fieldName>
      <expression>(!Map_Scale! / 2000)</expression>
    </CalculateValue>
</CalculateValues>
<CalculateValues>
    <CalculateValue>
      <query>Category = 1</query>
      <fieldName>MinPS</fieldName>
      <expression>0</expression>
    </CalculateValue>
    <CalculateValue>
      <query>Category = 1</query>
      <fieldName>SOrder</fieldName>
      <expression>( !Map_Scale!- !Date_On_Map!)</expression>
    </CalculateValue>
</CalculateValues>
```

Note that this feature is only supported for some commands. See the Multi-Commands column in the Predefined Command code list above.

Return Codes

All command operations or functions within MDCS system return either true or false to show the result of the last run operation. The client application can keep track of any operations that failed using the status codes returned. Further details on any errors/warning are available in the log files generated in the log folder where the standard GP error codes and descriptions can be found. Based on these predefined and standard error codes, a user can refer to the standard ArcGIS Help to find an acceptable solution to fix any errors found.

Reporting

MDCS includes a reporting system. For each execution of MDCS, a report file with a name of the form **ConfigFileNameYYYYMMDDT#####.xml** is written to the \logs\ directory, where "####" represents time of execution, so a unique log file is always created to ensure the result of a previous process is never overwritten.

Calling MDCS from Batch files

MDCS can be called by using batch files with commands of the following form.

python.exe MDCS.py <configuration file> <options>

A configuration file is the only mandatory input, as shown above, and the optional commands are listed above under **MDCS command line arguments**.

In this way multiple calls to MDCS can be chained together for more complex workflows involving the creation of multiple mosaic datasets.

Calling MDCS from Model Builder

MDCS and all its relevant library modules can be imported as standard python objects using code in an ArcGIS python toolbox. Once MDCS gets properly imported without any errors, all classes/tools defined within the toolbox can make calls to MDCS as a program object to access its functions. MDCS can also be run as an external process with valid command line arguments if required. This is akin to running MDCS from batch files with correct arguments.

ArcGIS users who would like to leverage MDCS within Model Builder to help build workflows could create Tools within a python tool box to associate MDCS configuration file content with relevant Model Builder UI elements on screen for editing. One or more editing tools can be created to edit the same configuration file, but at different places in the same file. This way, a tool can exist to edit Mosaic Dataset Properties in the configuration file while another could exist to edit process information.

Contents of configuration files for MDCS

As noted above, the recommended method for creating configuration files is to copy an existing file for a similar workflow and make edits appropriate for your new data. However, for developers and data managers seeking an indepth understanding of the configuration files, this section will provide further detail.

The master.xml file is a template that contains the choices and defaults for each parameter. Below is a guide to understanding these parameters and choices.

Understanding the Master.xml template.

The master .xml consists of several nodes and sub nodes. The main node name is *Application* which encompasses all the other sub nodes. The nodes correspond to the commands listed above. Whenever a command is required it is imperative to make sure that the corresponding node exists in the configuration file and is correctly populated. The list of major node names below is followed by detailed explanations of each.

1) Name
2) Command
3) ArcGISVersion
4) Workspace
    a. WorkspacePath
    b. Geodatabase
    c. MosaicDataset
    d. AddRasters
    e. CreateReferencedMosaicDataset
    f. DefaultProperties
    g. Table
    h. Functions
    i. Processes

1) Name

This corresponds to the name of the project. Edit this parameter to reflect the name of the project you are going to use the configuration file for. This helps in logging the actions of MDCS.

2) Command

This node takes in the predefined command codes specified above. The command codes correspond to specific GP functions. A list of the command codes are listed above in **Predefined Command Codes**.

3) ArcGIS Version

This node restricts the configurations scripts to run only with defined versions of ArcGIS. This is an optional node and can be omitted if your script can run with any version of ArcGIS.

Example:

```
<ArcGISVersion>
        <Product>
            <Min>10.2.0.3264</Min>      #(Major,  Minor, SP, Build (build value can be zero to ignore build check)
          <Max></Max>  #Optional
         </Product>
</ArcGISVersion>
```

Please also note, existing MDC Scripts will continue to work unaffected by the version check and it's up to the configuration creator to enforce the version check by adding/editing the respective values within the ArcGISVersion node.

4) Workspace

The workspace defines parameters related to creating, populating and setting properties of the mosaic dataset.

a) **WorkspacePath**: This is the path to the folder where the GDB will be stored. This workspace can also be used to define an SDE connection. An SDE connection can be defined either using an SDE filename (see Geodatabase below) or using a connection string.
For example: **c:\Image_Mgmt_Workflows\browseimagery\MD**
SDE example (Connection string): **Database Connections**
SDE example (file name): **c:\Image_Mgmt_Workflows\browseimagery\SDE**

b) **Geodatabase**: This is the name of the geodatabase. Typically, it is the name of the geodatabase in which you would like to create the mosaic dataset.
For example: **BIImage.gdb**
SDE example (Connection string & file name): **Connection to EG1146 (2).sde**

c) **MosaicDataset**: The mosaic dataset node contains properties required to create the mosaic dataset. Here you can specify the Mosaic Dataset Type, Name, SRS, Number of Bands and the Pixel Type. In the Master XML the nodes are filled with either a dummy name or valid choices. The Mosaic Dataset type is a parameter that is required by MDCS, with three valid types: Source, Derived, or Referenced. Refer to the Image Management Guidebook in the ArcGIS Help system for a detailed discussion of these three types. More information regarding the other parameters can be found here.

d) **AddRaster**: The add raster node contains information regarding the data source that needs to be added to the mosaic dataset. Replace the values that are in these nodes to point to the dataset that needs to be added. More help on this topic can be found here. **NOTE**: regarding the Raster Type, if the data will be loaded using the default type, "Raster Dataset" and all default settings, no further detail is required, but if the type or any properties must change, the user must create an *.art.xml file, and import that to define the Raster Type.

e) **CreateReferencedMosaicDataset:** The contents of this node are used when the command CR is used. A referenced mosaic dataset has limited capabilities, so only a limited number of operations can be performed on it. These operations include ERF, BB, SS, SP. More information about Referenced mosaic datasets can be found here.

f) **DefaultProperties**: Default properties control how the mosaic dataset can be used when it is being displayed or published. Edit the values appropriately and keep only the appropriate choices where multiple choices are defined.
For example: <resampling_type>**NEAREST;BILINEAR;CUBIC;MAJORITY**</resampling_type>, should include only one choice after editing, e.g.: <resampling_type>**NEAREST** </resampling_type> For more help on this topic please refer to this link.

g) **Table**: This node defines additional fields to be added to the mosaic dataset attribute table. In the master XML file you can include multiple fields. An example is given below. You will need to copy and paste additional nodes to define more than one field.

Add Single Field	Add Multiple Fields

<Fields> <Field> <name> **SampleFld**</name> <type>**TEXT**</type> <length>**32**</length> </Field> </Fields>	<Fields> <Field> <name>**SampleFld**</name> <type>**TEXT**</type> <length>**32**</length> </Field> <Field>
	<name> **SampleFld2**</name> <type>**LONG**</type> <length>**32**</length> </Field> </Fields>

For help regarding supported field types look here.

h) **Functions**: Enter the path to the Raster Function Template file. The raster functions allow you to define processing operations to the rasters in the mosaic dataset.

For a list of functions used by mosaic datasets go here.
For information on how to create a Raster Function Template file click here.

i) **Processes**: The processes defined in this node correspond to the various commands specified using the command line or the commands node in the configuration file. The parameters are only accessed by MDCS if the command is specified. Only edit the parameters of the nodes of the commands that are specified.

For example: Commands specified are CS+BPS. The configuration nodes that are read in by MDCS are *CalculateStatistics* and *BuildPyramidsAndStatistics.* Other configuration nodes may be populated in the configuration file, but they will be ignored.

For help regarding the individual parameters specified in this section please see table of command codes above. The parameters correspond to the parameters defined in the help. Its is generally best to use an existing template or extract the required node names and samples from the Master.XML

Example MDCS Project

This documentation on MDCS accompanies a separate document and supporting files which provide an example project. The example can be used with any imagery that matches the simple 'Raster Dataset' Raster Type – for example, georeferenced tiff files. To run that example or start a new project, refer to the example documentation.