



Planners ARM-ed and prosperous

The MineSight Interactive Planner (MSIP)-ARM utility was originally written for grade control use, but its benefits for planners and drill and blast engineers were quickly realized. ARM stands for Attribute, Release and Manage, and is an add-on to the popular MSIP. The product consists of three utilities (ipAttributer, ipRelease and ipManager) that are executed from the Scripts tab in MSIP.

Three main principles make MSIP-ARM such a flexible tool:

1. The first enables you to attribute cuts stored in the MineSight Planning Database (MSPD) based on values, calculations or wildcard tokens, such as date or user name. This attribution can be done automatically as a cut is made, run manually against each cut, or run against a whole IP-Set.
2. Secondly, you can release the information stored in the MSPD in any number of formats. For example, releasing a point file for survey or material attributed polygons for plotting.
3. The key principle behind MSIP-ARM however is the ipManager interface. This allows you to control the attribution and release without any assistance or scripting. It puts the power in your hands!

This article will focus on the attribution part of MSIP-ARM, which is distributed as a standard part of MineSight. The Release component of MSIP-ARM is part of the MineSight Axis-Grade Control package and does have a cost associated with it. The best way to understand it is to look at examples of how it has been used. The following are generic examples of how MSIP-ARM has been used at sites.

Tokens

Tokens are used in the ipManager to dynamically set values in the attribution and release parts of the utility. Tokens can be model items, built-in tokens or any user defined attribute. They are defined by a particular syntax `%(token name)s` where the `s` can be substituted to control the desired output. Examples of tokens and using different extensions are below.

Simple Attributions

Date/Time tokens can be used in any order to assign the current date to a cut (i.e. when it is created):

`%(d)02i%(m)02i%(y)02i` would return 280711

`%(d)02i-%(month)s-%(year)s` would return 28-July-2011

It is possible to set evaluation statements to further control the format of the date:

`#eval "%(d)02i-" + "%(month)s"[:3].upper() + "-%(year)s"` would return 28-JUL-2011

Complex Attributions

Executive statements can be used to perform more complex attributions on a cut. This would be done using MineSight's Grail scripting language, which is based on Python. For example, it is possible to extract part of the name of a geometry object and assign that to an attribute. The material name of a blast polygon could be used to set a BLAST attribute:

```

Calculation
1 #exec
2 from grail.data import geometry
3 #
4 ACTIVE_BLAST_MSR = r'C:\MSAxis\_msresources\Current\Blast.msr'
5
6 blastPattern = "UNKNOWN"
7 -if os.path.exists(ACTIVE_BLAST_MSR):
8     msr = geometry.Geometry(ACTIVE_BLAST_MSR)
9     blastPattern = msr.getpolygonnameat(0).upper()[-3:]
10
11 result = ("%s" % (blastPattern))

```

Simple Calculations

Tokens can be mixed with fixed values to generate calculated values:

%(tonnes)/1000 can be used to output Kilotonnes

%(tonnes)*%(grdAU)/31.103477 can be used to output gold ounces

The format of the results above can be controlled using executive statements. Executive statements can be multiline calculations or simple calculations with more complex syntax. To limit the output of a calculation to two decimal places, write:

```
#exec
```

```
Result = round(tonnes*AU/31.103477, 2)
```

Complex Calculations

MSIP-ARM is useful for complex calculations. It can access the underlying reserves data stored in the MSPD and provide a breakdown of grades and tonnes for each material type in a cut, which is very useful for short term planning. Below is an example of the syntax that calculates the gold grade of fresh material at the first cutoff in a Material Set. To obtain such detail, contact your local MineSight support office for help setting this up:

```

Calculation
1 #exec
2 from grail.ip import reslib
3 ##Configurable options
4 MATERIAL = ["Fresh"]
5 CUTOFF = [0]
6 GRADE = "AUKR" #Tonnes, Volume or Model Item name
7 MWA = not reslib.isgradeaccum(DRES, GRADE)
8 ##End of Configurable options
9 tonnes = 0.0
10 volume = 0.0
11 metal = 0.0
12 result = 0.0
13 -for reserve in DRES['cuts'][index]['reserves']:
14     -if reserve['material'] in MATERIAL and reserve['cutoff'] in CUTOFF:
15         tonnes += reserve['tons']
16         volume += reserve['volume']
17         -for gradeName, gradeValue in zip(reserve['gradenames'], reserve['gradevalues']):
18             -if gradeName == GRADE:
19                 metal += (reserve['tons'] * gradeValue)
20
21 -if GRADE == "Tonnes":
22     result = tonnes
23 -elif GRADE == "Volume":
24     result = volume
25 -elif MWA: #MWA grade
26     -if tonnes:
27         result = metal / tonnes
28     -else:
29         result = 0.0
30 -else: #accum grade
31     result = metal
32 result = round(result, 3)

```

Material Classing

Another way to assign a value to an attribute using MSIP-ARM is to use the Material Classing Grid to define a material type or destination based on multiple items and/or attributes. Material classing is like the MSBasis modcls.dat procedure.

This option allows you to assign the class value to two attributes. This lets you set an alpha code for display and a numeric code for writing back to a block model.

Add as many model items or attributes as you like to the grid, and for each row you can define the data boundaries.

You can use standard greater-than etc. symbols for numeric values, or if the number is an integer you can use '-' to define between or separate numbers with a comma. It is important with these types of grid to ensure that all material is classified. Do this by setting a default class at the bottom that will capture anything not set by the grid:

Material Classing					
Material Class Attribute 1		DEST	Material Class Attribute 2		OTN
	Class 1	Class 2	grdAU	grdSTOT%	majOXIDE
1	SHGHS	1	0.0 99.0 >=10	0.0 100.0 >=2	0.0 9.0 3
2	SHGLS	2	0.0 99.0 >=10	0.0 100.0 <2	0.0 9.0 3
3	HGHS	3	0.0 99.0 >=6.5 <10	0.0 100.0 >=2	0.0 9.0 3
4	HGLS	4	0.0 99.0 >=6.5 <10	0.0 100.0 <2	0.0 9.0 3
5	MG	5	0.0 99.0 >=5 <6.5	0.0 100.0	0.0 9.0 3
6	LG	6	0.0 99.0 >=2 <5	0.0 100.0	0.0 9.0 3
7	TRHG	7	0.0 99.0 >=6	0.0 100.0	0.0 9.0 2
8	TRMG	8	0.0 99.0 >=4.5 <6	0.0 100.0	0.0 9.0 2
9	TRLG	9	0.0 99.0 >=1.8 <4.5	0.0 100.0	0.0 9.0 2
10	OXHG	10	0.0 99.0 >=6	0.0 100.0	0.0 9.0 1
11	OXMG	11	0.0 99.0 >=4.5 <6	0.0 100.0	0.0 9.0 1
12	OXLG	12	0.0 99.0 >=1.8 <4.5	0.0 100.0	0.0 9.0 1
13	MW	13	0.0 99.0 >=1 <1.8	0.0 100.0	0.0 9.0 1
14	WST	14	0.0 99.0	0.0 100.0	0.0 9.0
15			0.0 99.0	0.0 100.0	0.0 9.0

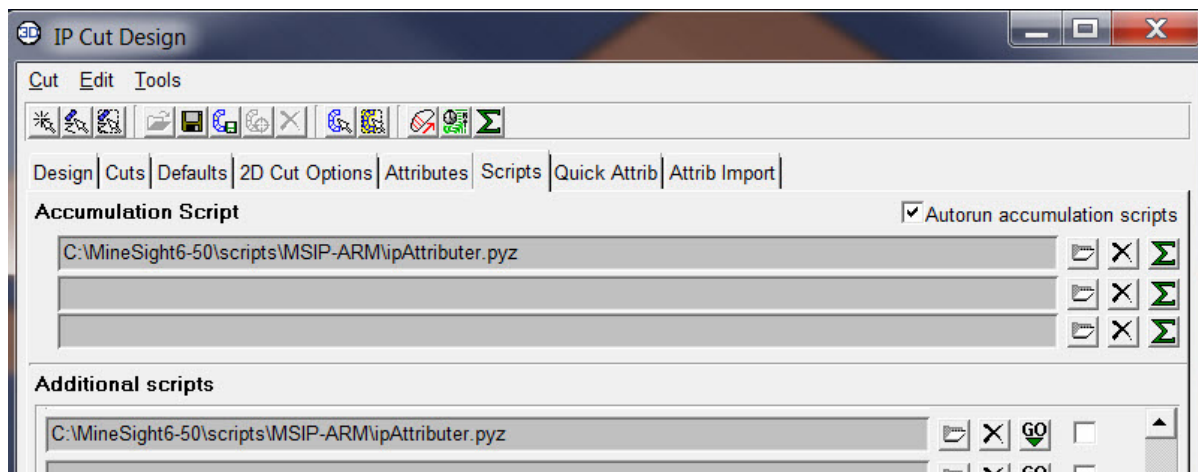
As a guide to setting the bins the cell shows the item min/max values and the bin chosen is shown in red.

It is possible to make multiple classing templates and run them sequentially so the results of the classing above could be used to define the material destination.

Material Classing			
Material Class Attribute 1		STKP	Material Class Attribute 2
	Class 1	Class 2	DEST
1	ROM-HS		SHGHS,HGHS
2	ROM-HG		SHGLS,HGLS,TRI
3	ROM-MG		MG,TRMG
4	ROM-LG		LG,TRLG
5	STK-HG		OXHG
6	STK-MG		OXMG
7	STK-LG		OXLG,LG,TRLG
8	MW		MW
9	WST		

Running Attributions

Usually, you would set the ipAttributer utility as an Accumulation Script in the IP-Tool and autorun it so that cuts get attributed on the fly as you make them. However, it is also possible to run attributions on all of the cuts in an IP-Set at once. This can be very useful when you have added a new attribute to an IP-Set and wish to set it automatically.



First, make a new ipManager .ini file from the dialog and define the attribution parameters. Save this with a specific name so you remember it.

Now you can run the ipAttributer from the Additional Scripts section. You will be warned that this will affect all the cuts in the IP-Set. Then it allows you to choose the .ini file to use.

MSIP-ARM has many more uses than are shown here. For more information, contact your local MineSight support office and give MSIP-ARM a try.