Maximo Formula's are the logical next step in Maximo customization after Maximo Scripting. Maximo formulas follow Excel like grammar to define expressions that use input from variables to calculate a value. Unlike scripting, where most of the variables need to get predefined and bound to some Maximo attributes/properties/maxvars, the formula expression can use any of those Maximo attributes/properties/maxvars inside the expression without ever needing to predefine or bind them.

Before we jump the gun and start thinking that formulas are replacement for Scripting, let's set the scope for it. The scope of Maximo formula's are limited to mathematical expressions and hence it can be only used to calculate numerical values. Also expressions are not full programming constructs. Automation scripting on the other hand leverage all the power that comes with dynamic scripting languages like JavaScript/Python. So while a powerful and quick way to get calculated values, formulas serve a niche purpose and are no replacement for Automation scripts.

A formula can be of associated either with a Mbo attribute or a Mbo

- Attribute Formula
- Object Formula

An attribute formula is very similar to an attribute launch point in Scripting world. It allows one to associate a formula with a mbo attribute. Effectively that mbo attributes value gets calculated using the formula. Generally calculated values are represented using non-persistent attributes in Maximo. For example we can define a non-persistent attribute called *overbudgetcost* in Asset to keep track of the amount by which the Asset cost year to date has gone over the budgeted cost.We can then write an attribute formula for *overbudgetcost* as ytdcost-budgetcost. This makes sure that whenever *ytdcost* or *budgetcost* changes, the *overbudgetcost* value changes too. It also makes sure that when the Asset Mbo gets loaded, the *overbudgetcost* gets calculated. The formula evaluator also understands the dependency of *overbudgetcost* to *ytdcost* and *budgetcost* and hence whenever *ytdcost* or *budgetcost* changes, the *overbudgetcost* value changes too without needing to do anything else. Such a thing would have needed few more launchpoints (one for ytdcost and one for budgetcost in additon to the one for overbudgetcost) to be defined in Automation Scripts to get this dependency implemented.

Attribute Formula’s can also be associated with persistent attributes. In the above case if *overbusgetcost* was a persistent attribute, the formula is evaluated at save point of the Mbo as opposed to the init of a Mbo (as in case of non-persistent attributes). We will talk about asynchronous evaluations of formulas associated with persistent attributes later in this document.

An object formula is a formula that is not associated with any Mbo attribute but is just associated with the Mbo itself. These are often known as dynamic non-persistent attributes because these expressions can be evaluated for a Mbo and their values used as input to other formulas, just

like you would use an attribute value in other formulas. The difference being, we do not need to explicitly create these Mbo non-persistent attributes (using DB config app+Admin mode etc) just to store a calculation. If say we defined that overbudgetcost as a formula instead of a non-persistent Mbo attribute, we could have used that as part of another formula like below

exp$overbudgetcost + someothermboattr

The key thing here is, we can use the key word “exp$”, append the object formula name to it and then use it as part of any other calculation. Similarly you can use it as part of

Maximo formulas support most of the well known mathematical operators and functions and marries the expression grammar with the Maximo Mbo’s and mbo attributes.

A list of the supported operators and functions are shown below

Operators

Operator	**Description**	**Usage**
+	Addition	purchaseprice + tax
-	Subtraction	enddate - startdate
*	Multiplication	unitcost*quantity
/	Division	totalcost/quantity
%	Remainder	a%b
^	Pow	a^2
&&	Logical And	IF(a>10 && b<20,x,y)
||	Logical Or	IF(a>10 || b<20,x,y)
>,>=	Greater than,Greater than equals	IF(a>=10 || b<20,x,y)
<, <=	Less Than,Less than equals	IF(a>10 || b<=20,x,y)
=,==	Equals comparison	IF(a=10 || b==c,x,y)
!=,<>	Not equals	IF(a<>10 && b!=c,x,y)

Functions	Number of Params	Description	Usage
IF	3	Used for If conditions. Takes in 3 parameters, the condition, the true value and the if false value. In the example shown the first parameter is the condition which if evaluates to true, the IF function will have a value of the variable x. If the condition is false, the IF function will have a value of the variable y. As obvious these x and y can inturn be nested functions.	IF(a<>10 && b!=c,x,y)
MAX,MIN	2	As the name suggests, these functions are used to calculate the max and min values between 2 numbers. One can use nested calls to these functions to support a max/min of more than 2 numbers.	MAX(a,b) or MAX(a,MAX(b,c)) MIN(a,b) MIN(a,MIN(b,MIN(c,d,e)))
ABS	1	The absolute value of a number.	ABS(a)
DATE	3	A Date in millis with year,month,day,	DATE(1994,10,12) is 12th of October in 1994.
DATETIME	6	A date time in millis with year,month,day,hour,mins,sec. The hour follows the 24 hr format.	DATETIME(1994,10,12,14,0,0) is 2pm in 12th of October 1994.
DURATION	6	The duration function takes in 6 parameters - DURATION(year,month,day,hour,min,sec) - to specify a duration in millis.	DURATION(0,0,0,2,0,0) is a duration of 2 hours.
ROUND	2	Round a number based on a precision/scale	ROUND(linecost,2)
NVL	2	Returns the alternate value if the original value is null.	NVL(a,b) - returns b is a is null.
PCT	2	Returns the percentage of the first param value based on the 2nd param value.	PCT(val,total)

FLOOR,CEILING	1	Floors and ceils the value.	FLOOR(3.2) will return 3.
SQRT,SIN,LOG,LOG10,TAN,COS,ASIN,ACOS,ATAN,SINH,COSH,TANH,RAD,DEG	1	Basic math functions as their name suggests.	SQRT(4) returns 2.

In the usage samples, all the variables ie the *a* and *b*'s would be replaced with Maximo artifacts. A maximo artifact would be

- A mbo attribute.
- A maxvar.
- A maximo property
- A maximo condition.
- Another expression.
- Literal value.

Each formula is executed in the context of a Mbo. A mbo attribute in the expression would refer to an attribute or a related attribute of that Mbo. An example below describes the usage of each in context of Asset mbo. Say we want to calculate the health score (range of 1..3) for an asset based on its pressure meter value. The formula below can be used to do that.

IF(pressuremeter$lastreading>60, 1, IF(pressuremeter$lastreading>20 && pressuremeter$lastreading<59, 2, 3))

Key things to note here is:

1. We created a relationship from ASSET to ASSETMETER for pressure meter called PRESSUREMETER. A sample such relation where would like like this below.

    assetnum=:assetnum and siteid=:siteid and metername='O-PRESSUR'

2. We then used the relation names and the attribute name (lastreading) that holds the value of the last meter reading to make the expression. Note that instead of the dot notation <relation>.<attribute>, we use the $ as the separator. This is because the grammar of the Maximo formula's do not allow a "." separator.
3. The IF function is nested to cover all ranges of pressure values.
4. All attributes and related attributes used in the formula would be converted to decimal values for calculation. In this case, lastreading was of type ALN and hence the formula evaluator would convert that to a decimal for evaluating the expression. For example a value

like “45.25” would be converted to a decimal 45.25. All dates would be converted to millis for calculation. All booleans would be treated as 1 (true) or 0(false).

Now that we have seen a Mbo we can deal with a few of the other types of supported variables before jumping into some of the more advanced concepts with variables.

A sample with a maximo property and maxvar is shown below

prop$*propname**10 + var$*varname*

Is a simple expression that uses the property value of *propname* and maxvar value of *varname* in the expression. One caveat though is the property/maxvar name cannot contain certain characters (like the “dot” character) as those are not allowed in the formula grammar.

MAX, MIN, AVG, SUM and COUNT

Maximo formula’s allow evaluating the count,max,min,avg and sum based on related set of Mbos. An example below shows those for a Purchase Order (PO) Mbo calculating those for the related POLINE’s.

sum$poline$linecost	this calculates the sum of linecost’s for all POLINE’s for the PO. Syntax sum$<relation name>.<attr name>
count$poline	this calculates the total number of POLINE’s for the PO. Syntax count$<relation name>
avg$poline$linecost	this calculates the average cost per POLINE for the PO. Syntax avg$<relation name>.<attr name>
max$poline$linecost	this calculates the maximum cost for all POLINE’s for the PO. Syntax max$<relation name>.<attr name>
min$poline$linecost	this calculates the minimum cost for all POLINE’s for the PO. Syntax min$<relation name>.<attr name>

An example formula leveraging these is shown below for the Mbo Asset:

IF(count$openwo>0,0,1)

Which implies if the count of open workorders for an asset is more than 0, then the expression evaluates to 0 and it evaluates to 1 otherwise. In here openwo is a relationship name from asset to workorder.

Often times you would want to evaluate these in context of a duration. For example you might want to get the max meter reading in last 7 days. That can be achieved using the function equivalent of these expressions.

FMAX$pressuremeasurement$measurementvalue$measuredate(DURATION(0,0,7,0,0,0))

If we look at this expression carefully, you would notice 4 tokens separated by $. The first one FMAX is the name of the function. The 2nd one is the name of the relationship from ASSET to MEASUREMENT table (as pressure is of type gauge, Maximo would store the values in measurement table). The 3rd one being the attribute name whose MAX we want to calculate, which in this case is measurementvalue. The 4th one being the date attribute based on which we will calculate the last 7 days duration.

Note if we use a value of -1 here instead of DURATION, then this just calculates the MAX for all values fetched by that relationship. This then becomes the semantic equivalent to the max$ notation.

FSUM$pressuremeasurement$measurement value$measuredate(DURATION(0,0,7,0,0,0))	this calculates the sum of measurementvalues for that asset meter for the last 7 days (pivoted around measuredate)
FAVG$pressuremeasurement$measurement value$measuredate(DURATION(0,0,7,0,0,0))	this calculates the average of measurementvalues for that asset meter for the last 7 days (pivoted around measuredate)
FMAX$pressuremeasurement$measurement value$measuredate(DURATION(0,0,7,0,0,0))	this calculates the maximum of measurementvalues for that asset meter for the last 7 days (pivoted around measuredate)
FMIN$pressuremeasurement$measurementv alue$measuredate(DURATION(0,0,7,0,0,0))	this calculates the minimum of measurementvalues for that asset meter for the last 7 days (pivoted around measuredate)

There is no count equivalent for these functions yet. It will be added soon.

Evaluating conditions

Evaluating maximo conditions are often necessary as the conditions in the IF function may often involve non-numeric values. For example if we wanted to write an expression that would provide a numeric value for each of the possible oil colors in a characteristic meter we would have needed to write a formula expression like below.

IF(oilmeter$lastreading==”CLEAR”,5,IF(oilmeter$lastreading==”LIGHTBROWN”,4,IF(oilmeter$lastreading==”BROWN”,3,IF(oilmeter$lastreading==”DARKBROWN”,2,IF(oilmeter$lastreading==”RED”,1,0)))))

But unfortunately comparing ALN values is not what the formula evaluator can support right now. So comparing the lastreading (ALN) value to “CLEAR” would be a syntax parsing error. The formula expression below however would do the job correctly.

IF(cond$OILCLEAR,5,IF(cond$OILLBROWN,4,IF(cond$OILBROWN,3,IF(cond$OILDBROWN,2,IF(cond$OILRED,1,0)))))

In the above example we have defined 5 conditions in the maximo conditon expression manager application - OILCLEAR, OILLBROWN, OILBROWN, OILDBROWN and OILRED.
A sample condition text (for the OILCLEAR condition) is show below :

oilmeter.lastreading=’CLEAR’

Now notice the key word cond$ which we prepend to the condition name for the formula evaluator to understand how to evaluate it. The formula evaluator will evaluate that to a 1 or 0 based on whether the condition evaluates to true or false. So you can now follow how this will be evaluated - if the oil color is CLEAR it will evaluate to 5 if not it will go to the nested IF function to check if the oil color is BROWN and so on. Effectively we have now a numeric score for each of the oil colors in an asset meter with a max score (color is CLEAR) of 5 and a min score (when say color is TURBID) of 0.

As you might have guessed, the conditions can be any valid condition that Maximo allows, including SQL expressions.

Checking for Null

A lot of times checking for null might be needed for evaluating formula expressions. A simple way to do that in Maximo formula’s is illustrated below
IF(isnull$installdate,...)

This checks if the install date in ASSET is null and based on that do some expression evaluations. Of course there is a isnotnull$<attribute name>. As you might have guessed the attribute name can be a related attribute like isnotnull$location$classatructureid would check if the related locations (from ASSET) classstructureid is set or not.

In some cases, we can avoid these explicit null checks to provide an alternate for a null attribute value can be easily achieved using the NVL function.

Working with previous values in Mbo Attributes

As in mbo attributes, it often makes sense to work with previous values of a Mbo attribute. Maximo formula's provides an easy way to access them using the expression below

IF(prev$priority>priority,...)

Which implies if the priority of the ASSET has be down-graded evaluate some expression. The attribute name can be a related one too.

In the same line, the formula grammar provides a way to check if the attribute has been modified. A sample is shown below

IF(modified$priority,...)

Working with latest, oldest and last but one values in a related MboSet

We can access the latest or the oldest values from a related set of Mbos and use them in the formula expressions. An example below shows how to calculate the install date of an Asset if one is not present in Maximo asset record.

NVL(installdate,oldest$allwo$statusdate - DURATION(0,6,0,0,0,0))

This uses the NVL function to check if the installdate is not null, use that, otherwise use the allwo relation to get list of all workorders and find the oldest workorder by statusdate and use that oldest statusdate value (munus a duration of 6 months) as the approximate install date for the Asset.

Another example say would be to use the latest open workorders (for an asset) priority for an expression. An example is shown below.

NVL(latest$openwo$wopriority$statusdate,2)

The expression latest$openwo$wopriority$statusdate gets the wopriority attribute value of the latest open workorder (using relationship openwo from ASSET) by statusdate. The NVL around it would evaluate this value to 2 is the worpriority is null.