cmlmcol: Code Magus Metric Collector Version 1

CML00093-01

Code Magus Limited (England reg. no. 4024745)
Number 6, 69 Woodstock Road
Oxford, OX2 6EY, United Kingdom
www.codemagus.com

December 15, 2020

Contents

1 Introduction

This document describes the interface to the Code Magus Limited `cmlmcol` server.

All Client requests for metrics are via the command interface of `cmlmcol`.

2 Command interface

2.1 Command Elements

The elements of the commands to `cmlmcol` comprise reserved words, identifiers, string literals, comments and integers. The commands are free format and white spaces have no grammatical meaning except where they might appear within string literals.

2.1.1 Comments

Comments are introduced by using a hash ('#') and continue up to the end of the current input line.

2.1.2 Reserved Words

Reserved words have a special meaning in terms of directing the parsing of commands. Please note that reserved words are not case sensitive. The reserved words are:

2.1.3 Identifiers

An *Identifier* is case sensitive, it starts with a letter which can be followed by any number of letters, digits or the under-score character.

Examples:

```
ncacrag_123   RecordStaffArrgmntDet
```

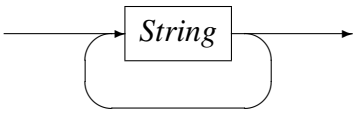
2.1.4 Strings

Strings are:

- any sequence of characters (except double quotes and the newline character) enclosed by double quotes.

- any sequence of characters (except single quotes and the newline character) enclosed by single quotes.

Strings cannot span source text lines, but they may be concatenated:



Examples:

```
"GigabitEthernet0/0 In Octets"
'$Revision: 1.1 $'
```

2.1.5 Integers

A *Integer* consists of a nonempty sequence of decimal digits '0' through '9'.

Examples:

```
1234
0
```

2.1.6 Numbers

A number consists of a nonempty sequence of decimal digits that

- possibly contains a radix character (decimal point '.').
- is optionally followed by a decimal exponent; consisting of an 'E' or 'e' followed by an optional plus or minus sign followed by a nonempty sequence of decimal digits that indicates multiplication by a power of 10.

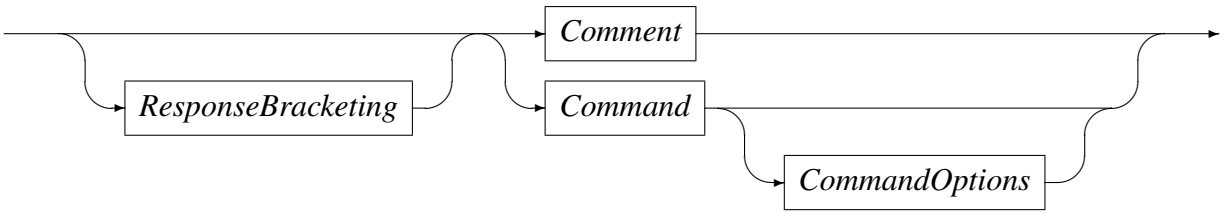
Examples:

```
1234
0.001
1.2
123.45E-12
```

2.1.7 TimeStamp

The *Timestamp* have the following format `CCYYMMDDhhmmss`.

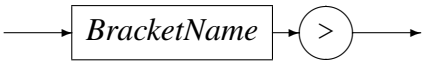
2.2 General Command Syntax



Input to the command processor is either:

- A *Comment*. The whole line is ignored by the command processor, see 2.1.1 on page 2.
- A *Command*, optionally followed by command options.

ResponseBracketing



BracketName can be any character except the '>' character. The response to the `cmlmcol` command will be preceded by '`.begin BracketName>`' and followed by a newline and '`.end BracketName>`'.

Example:

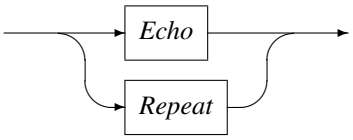
```
my context> get value metric(a.b.c) function(rate)
```

Response:

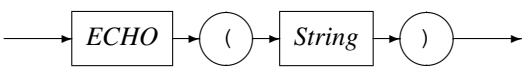
```
.begin my context>
.
.
.
.end my context>
```

All commands can be followed by zero or more of the command options. These options affect the way in which the command is executed; for example causing the command to be repeated at intervals.

CommandOptions



Echo



Causes the echo option as specified to be appended to the command response.

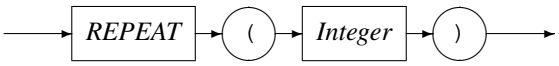
Example:

```
get value metric(a.b.c) function(rate) echo("my context")
```

Response:

```
.
.
.
echo("my context")
```

Repeat



Repeat the request every *Integer* seconds.

Example:
Repeat the request every 30 seconds.

```
get value metric(a.b.c) function(rate) repeat(30)
```

Repeat the request every 30 seconds with echo.

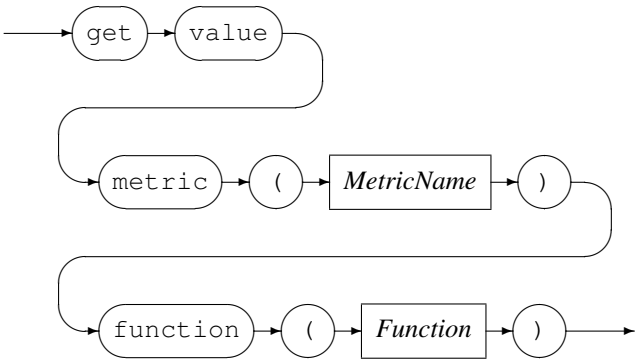
```
get value metric(a.b.c) function(rate) echo("my context") repeat(30)
```

2.3 List Command

2.4 Get Value Command

Responds with the value for the requested metric using the calculation as per the function specified.

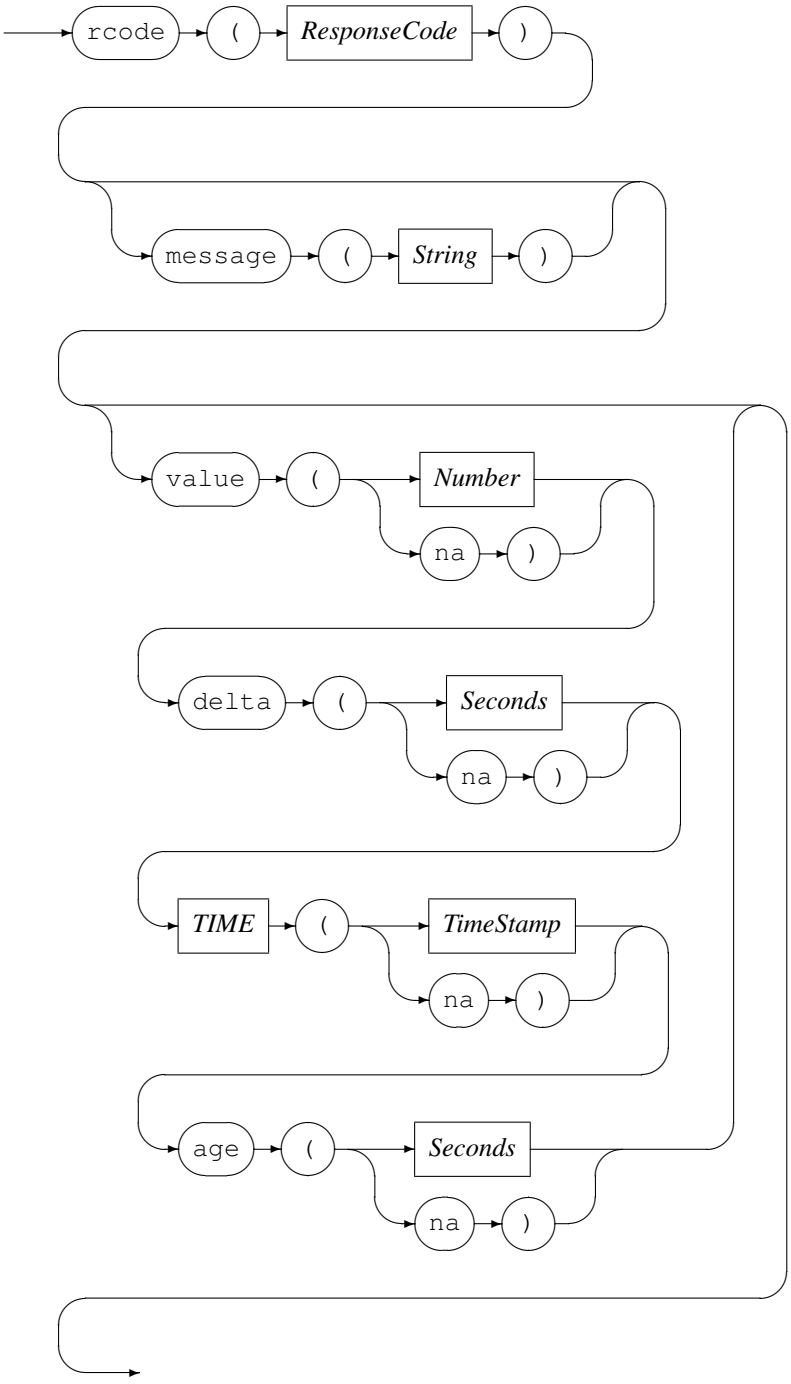
2.4.1 Request



For *MetricName* refer to appendix A on page 8

For *Function* refer to appendix C on page 9

2.4.2 Response



- *ResponseCode*

Completion code of the request. see appendix B on page 8

- `value`
  The calculated value using the requested function. Note if the value could not be calculated it contains the key word `na`.
- `delta`
  For the requested metric value, this is the time in seconds from the previous metric update received to the last update received by `cmlmcol`. Note if there was no previous update `delta` contains the key word `na`.
- `time`
  This is the *TimeStamp* of the last update received by `cmlmcol` for the requested metric. Note if the there was no updates `time` contains the key word `na`.
- `age`
  This is the seconds lapsed since the last update received by `cmlmcol` for the request metric. Note if the there was no updates `age` contains the key word `na`.

2.4.3 Examples

Get the value for `posipcon.prod_posipcon_00.session` using the function `last_5minutes_rate`. Request:

```
get value metric(posipcon.prod_posipcon_00.session)
   function(last_5minutes_rate)
```

Assuming the metric is defined and the function is a valid function, the response could be something like:

- Success full:

  ```
  rcode(0) value(3.053) delta(10) time(20131122094450) Age(7)
  ```

- Exception - no previous metric:

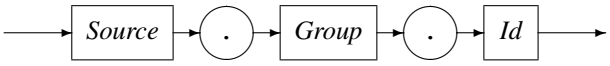
  ```
  rcode(1) message("No previous metric")
     value(na) delta(na) time(20131122094450) Age(7)
  ```

- Exception - no updates received:

  ```
  rcode(1) message("no updates")
     value(na) delta(na) time(na) Age(na)
  ```

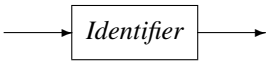
A Metric Name

A metric name is made up from three components:

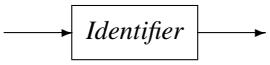


Source

Group



Id



B Response codes

Code	Description
0	Request success full
1	Request success full, with exceptions.
8	Request have semantic errors.
9	Request have syntax errors.

C Functions

Function	Description
delta_rate	interval rate
delta_average	interval average
smoothed_rate	smoothed rate
smoothed_average	smoothed average
last_5minutes_rate	last 5minutes rate
last_5minutes_average	last 5minutes average
current_count	count
current_sum	sum of values
current_sumsq	sum of the square of values
previous_count	previous count
previous_sum	previous sum of values
previous_sumsq	prev sum of the sq of values
delta_seconds	interval in seconds
delta_count	interval count
delta_sum	interval sum of values
delta_sumsq	interval sum of the square
last_5minutes_seconds	interval in seconds
last_5minutes_count	interval count
last_5minutes_sum	interval sum of values
last_5minutes_sumsq	interval sum of the square
last_hour_seconds	interval in seconds
last_hour_count	interval count
last_hour_sum	interval sum of values
last_hour_sumsq	interval sum of the square
start_of_day_count	
start_of_day_sum	
start_of_day_sumsq	
this_day_count	
this_day_sum	
this_day_sumsq	
this_day_sumsq	
previous_day_count	
previous_day_sum	
previous_day_sumsq	
start_of_hour_count	
start_of_hour_sum	
start_of_hour_sumsq	

Function	Description
this_hour_count	
this_hour_sum	
this_hour_sumsq	
previous_hour_count	
previous_hour_sum	
previous_hour_sumsq	
start_of_minute_count	
start_of_minute_sum	
start_of_minute_sumsq	
this_minute_count	
this_minute_sum	
this_minute_sumsq	
previous_minute_count	
previous_minute_sum	
previous_minute_sumsq	