In order to capture and replay application execution history for any tool, we need to add wrapper and/or change that tool's runner script.

Get installed cvl-emap tools list:

```
$ rpm -qa | grep -i '^cvl'
cvl-mesalib9_1-9.1-4.0.0.x86_64
cvl-openmpi1_6_1-1.6.1-4.0.0.x86_64
cvl-sdm_1d_calculate2_0_1-2.0.1-4.0.x86_64
cvl-levelset0_0_2-0.0.2-4.0.x86_64
cvl-emap-galaxy-shortcut0_0_1-0.0.1-4.0.x86_64
cvl-nn0_2_3-0.2.3-4.0.x86_64
cvl-3depict0_0_13-0.0.13-4.0.x86_64
cvl-keyutility1_0-1.0-4.0.3.x86_64
cvl-cvl-massive-preferences0_0_1-0.0.1-4.0.0.x86_64
cvl-virtualgl2_3_1-2.3.1-4.0.1.x86_64
cvl-hpc1_0-1.0-4.0.3.x86_64
cvl-posgen0_0_1-0.0.1-4.0.x86_64
cvl-effoff0_2_0-0.2.0-4.0.x86_64
cvl-emap-wiki-shortcut0_0_1-0.0.1-4.0.x86_64
cvl-caw0_2_3-0.2.3-4.0.x86_64
cvl-sdm_2d_calculate2_0_1-2.0.1-4.0.x86_64
cvl-imagej1_46-1.46-4.0.8.x86_64
cvl-logkeys0_1_1a-0.1.1a-1.0.x86_64
cvl-turbovnc1_2-1.2-4.1.4.x86_64
cvl-sdm_2d_plot0_0_1-0.0.1-4.0.x86_64
cvl-posminus0_2_2-0.2.2-4.0.x86_64
cvl-3daprecon0_0_1-0.0.1-4.0.x86_64
cvl-crystallography0_0_1-0.0.1-4.0.x86_64
cvl-xrngeditor0_0_1-0.0.1-4.0.x86_64
cvl-sdm_1d_plot0_0_1-0.0.1-4.0.x86_64
cvl-modules3_2_9-3.2.9-4.0.1.x86_64
cvl-visbenchmark1_0-1.0-4.0.0.x86_64
cvl-shellscripts0_1_0-0.1.0-1.0.x86_64
cvl-libjpeg-turbo1_3_0-1.3.0-4.0.0.x86_64
cvl-cvl1_0-1.0-4.0.5.x86_64
cvl-rdf-kd0_0_1-0.0.1-4.0.x86_64
cvl-objexport0_0_1-0.0.1-4.0.x86_64
cvl-fouriertransform0_2_2-0.2.2-4.0.x86_64
cvl-mrf0_2_1-0.2.1-4.0.x86_64
cvl-r-launcher0_0_1-0.0.1-4.0.x86_64
cvl-matlabr2012b-r2012b-3.2.x86_64
cvl-pysshfs0_1-0.1-4.0.0.x86_64
```

We will use a sample application: MRF tool
Package: cvl-mrf0_2_1-0.2.1-4.0.x86_64

Step 1: Find out desktop menu item from which user clicks to run this tool. Note there might be more than one desktop menu items for a given package. If they have different `"Exec="` line, wrap them separately; If there is no desktop menu item at all, you don't need to do anything – it's not a real tool, it might be a shared library. Only wrap tools which you can run it from desktop menu:

```
$ rpm -ql cvl-mrf0_2_1-0.2.1-4.0.x86_64 | grep -i '\.desktop$'
```

Step 2: Open the desktop entry file(s): `/usr/local/share/applications/cvl-mrf-0.2.1-MRF-launcher.sh.desktop`, find the `"Exec="` line:

```
Exec=/usr/local/mrf/0.2.1/bin/MRF-launcher.sh-run
```

Step 3: Run the tool as usual, when it asks you input, just leave it and go to another terminal

window to find out the whole process tree:

```
$ ps -f -U <username> | grep 'MRF-launcher.sh-run'
```

Here is a example output of this command:

```
<username>     6206     1  0 13:39 ?        00:00:00 /usr/bin/gnome-terminal
-x /usr/local/mrf/0.2.1/bin/MRF-launcher.sh-run

<username>     6208  6206  0 13:39 pts/2    00:00:00 /bin/sh
/usr/local/mrf/0.2.1/bin/MRF-launcher.sh-run

<username>    12838  2759  0 15:04 pts/1    00:00:00 grep MRF-launcher.sh-run
```

Step 4: From the output, you can see our `MRF-launcher.sh-run` is run by a `gnome-terminal`, and the first process for our job is `6206`:

Step 5: Now we have the parent's PID, we can find out whole process tree of our tool

```
$ pstree -a 6206

gnome-terminal -x /usr/local/mrf/0.2.1/bin/MRF-launcher.sh-run
  ├─MRF-launcher.sh /usr/local/mrf/0.2.1/bin/MRF-launcher.sh-run
  │   └─cvlvisrun /usr/local/bin/cvlvisrun MRF-launcher.sh
  │       └─MRF-launcher.sh /usr/local/mrf/0.2.1/bin/MRF-launcher.sh
  │           └─MRF
  ├─gnome-pty-helpe
  └─{gnome-terminal}
```

Step 6: From the output we know that our MRF (The last MRF in the tree) tool is started by `/usr/local/mrf/0.2.1/bin/MRF-launcher.sh`, and this is the launcher we can safely add wrapper

Step 7: Add wrapper to tool

Open `/usr/local/mrf/0.2.1/bin/MRF-launcher.sh`:

```
#!/bin/bash

DIR="$( cd "$( dirname "${BASH_SOURCE[0]}" )" && pwd )"

$DIR/MRF


echo "Sending usage statistics"

read -p "Please hit [enter] to exit"
```

Change it to:

```
#!/bin/bash


# cvl_emap shell functions

. /usr/local/share/cvl_emap_functions
```

```
DIR="$( cd "$( dirname "${BASH_SOURCE[0]}" )" && pwd )"

log_start

script -q -c "$DIR/MRF 2>'$stderr_file_path'" -t 2>"$timing_file_path"
"$stdio_file_path"

log_stop
```

After test this wrapper, you can replace this launcher in the tool's source code, build and distribute rpm package. This is covered by CVL Emap Support document. In “5.3 SUBMITTING RPMS TO THE CVL REPOSITORY”