ActEV19 SDL Evaluation CLI Programming Primer
July 25, 2019

Here are the necessary steps to go through in order to build a deliverable form of your ActEV system using the Abstract ActEV Evaluation CLI and test it.

There are several steps to completing this process:

1. Git fork and Git clone the ActEV Eval CLI,
2. Adapt/modify the function implementations for your system,
3. Test your implementation,
4. Update your CLI with the latest abstract CLI code,
5. Complete Final Delivery Checks on a new Ubuntu installation,
6. Deliver your ActEV CLI implementation.

Step 1: Git fork and Git clone the ActEV Eval CLI

Go to the diva evaluation CLI repository by clicking on the following URL:
https://gitlab.kitware.com/actev/diva_evaluation_cli

Click on the “Fork” button to make a copy of the repository in your own space.

Git clone your repository to make changes. 2 relevant branches are available:

- *master*: contains the abstract CLI to implement, you should start from this branch.
- *baseline_system_master*: an implementation example to help you.

Advice:

- *You should execute a ‘git pull’ command occasionally to get the latest updates from the forked repository.*
- *You should use the master branch to start your implementation*

Step 2: Adapt/modify the function implementations for your system

The CLI contains three directories namely:

- *bin/*: a NIST-maintained directory composed of commands parser and some commands implemented by NIST. Users should not modify the contents of this directory
- *src/*: a User-maintained directory that will include your CLI implementation.
- *container_output/*: set of directories containing your own system output on the validation data sets.

src includes a directory named “entry points” that contains functions that has to be completed. Each command is associated to a specific entry point and calls this one during the command execution. Generally, entry point functions are only used to call a script that really handles the request. These scripts can be put inside the *src* directory as well. See the baseline system master branch to have more information. Moreover, *bin* directory should not be modified as it is maintained by NIST.

container_output is a directory containing at least one generated output for at least the most recent validation dataset. By default, it contains the output and the chunk file generated by the baseline system as well as the file index and the activity index of the validation set used. The naming convention is:

<Dataset>_<File>.json

Where:

- *<Dataset> :== one of published data sets. E.g.,* ActEV-Eval-CLI-Validation-Set3
- *<File> :== "file", "activity", "chunk", "output"*

Step 3: Test your implementation

Once you want to test your work, you can simply install the CLI by using the following commands:

```
$ cd diva_evaluation_cli
$ diva_evaluation_cli/bin/install.sh
$ actev -h
```

The last command should give the help of the CLI if it is well installed.
As the CLI is installed as an editable package, you do not need to reinstall it after a modification.

Execute:
- a specific command to debug it(e.g. *actev system-setup*)
- *actev system-setup* following by *actev exec* with the right arguments to run every step in order to get an output result.
- actev validate-execution to compare your result with the data set results included in *container_output.*

Advice:
execute actev validate-system often to make sure that your implementation is still valid with what NIST expects. This involves:
- *Entry point method signatures must not be modified. You cannot add, modify or remove a parameter in the signature.*
- *Structure of the project must not be modified. You cannot modify /bin.* You may write your implementation under */src* and some additional outputs in */container_output.*
- *Some expected outputs are provided in the /container_output/<Dataset>* directory. The files within each data set directory are the following: <Dataset>_file*.json, <Dataset>_activity.json, <Dataset>_chunk.json and <Dataset>_output.json.*
    - The output that you put in container_output should be consistent with the system output for Activity Detection (AD).
    - For example, the file listing for ActEV-Eval-CLI-Validation-Set3 is:
      ```
      /container_output/
      /container_output/ActEV-Eval-CLI-Validation-Set3
      /container_output/ActEV-Eval-CLI-Validation-Set3/ActEV-Eval-CLI-Validation-Set3_activity.json
      /container_output/ActEV-Eval-CLI-Validation-Set3/ActEV-Eval-CLI-Validation-Set3_chunk.json
      /container_output/ActEV-Eval-CLI-Validation-Set3/ActEV-Eval-CLI-Validation-Set3_file.json
      /container_output/ActEV-Eval-CLI-Validation-Set3/ActEV-Eval-CLI-Validation-Set3_output.json
      ```

execute actev system-setup should not require any manual action and should be fully automated.

Step 4: Get update from the forked repository and commit your changes to yours

In order to get the latest updates from the forked repository, you can define a remote upstream:

```
$ git remote add upstream https://gitlab.kitware.com/actev/diva_evaluation_cli
```

Once is done, execute the following commands to update your master branch:

```
# Fetch all the branches of that remote into remote-tracking branches,
# such as upstream/master:

$ git fetch upstream

# Make sure that you're on your master branch:

$ git checkout master

# Rewrite your master branch so that any commits of yours that
# aren't already in upstream/master are replayed on top of that
# other branch:

$ git rebase upstream/master
```

Commit your changes into your repository is simpler and only requires a *git commit* command followed by a *git push*.

Step 5: Complete Final Delivery Checks on a new Ubuntu installation

Complete these steps before delivering your implementation to NIST:

1. Perform Step 4 (Get CLI updates) Above
2. Test that you can re-generate a valid system output:
   a. On a fresh Ubuntu installation, run your system-setup command and check that your environment sets up properly.
   b. Run ‘actev-exec’ on the ActEV-Eval-CLI-Validation-Set3 data set outputting the data to container_output/ActEV-Eval-CLI-Validation-Set3_<TeamName> and add the other expected files (file, activity and chunk). See ‘Advice’ part section 3 to have more information. This directory has to be part of the delivery.
   c. Run the following command to compare the baseline system output to the produced output:

```
$ actev validate-execution \
-o \
container_output/ActEV-Eval-CLI-Validation-Set3_<TeamName>/ActEV-Eval-CLI-Validation-Set3_<TeamNa
me>_output.json \
-r \
container_output/ActEV-Eval-CLI-Validation-Set3_<TeamName>/ActEV-Eval-CLI-Validation-Set3_output.json
\
-a \
container_output/ActEV-Eval-CLI-Validation-Set3_<TeamName>/ActEV-Eval-CLI-Validation-Set3_activity.jso
n \
-f \
container_output/ActEV-Eval-CLI-Validation-Set3_<TeamName>/ActEV-Eval-CLI-Validation-Set3_file.json \
-R \
~/results
```

3. Pass checks on the system delivery by executing ‘actev validate-system’
4. Make sure that your system can run after a reboot of your machine (by including environment variable in your .bashrc)

Step 6: Deliver your ActEV CLI implementation
Once you have passed Step 5, you can deliver your system via URL:

Sign in to access the web platform (https://actev.nist.gov/), click on “ActEV SDL Evaluation” and create a new system. Once it is done, submit a URL pointing to your CLI implementation as well as the credentials (username, password and/or token) to access it so that NIST can download it, install it on a new Openstack instance and test it.

Currently, NIST supports git repository or web accessible compressed tar/zip file. Moreover, this option implies that your ‘actev system-setup’ command implementation is able to install all your package dependencies on a fresh Ubuntu system without any manual assistance.

What happens when I trigger the validation pipeline?

A new Ubuntu VM is created and your CLI implementation is downloaded and executed alongside with the last CLI implementation from NIST. More details about the execution of the commands can be found in the repository of the CLI. If your system processes the validation dataset successfully, the VM is saved as a *snapshot* and kept for further analysis in a sequestered environment.

Change Log:

- June 19, 2019 - Revised for SDL
- July 25, 2019 - Identified Validation Set 3 as the validation data set