**CACIE Tool #00** – ***Tool Runner (runner.py)***

**Version** **1.0**

**QA**: **TEST** or **NA** or **QA**

1. **Description and Purpose**

One or two paragraphs describing the tool’s function and purpose.

The Tool Runner acts as a controller for all the other utilities contained within the CA/CIE Tools set. The Tool Runner determines configuration control status of the invoked tool and documents its version and QA status (whether or not the invoked tool has been tested and qualified in compliance with the Integrated Software Management Plan).

1. **Functional Requirements**

The functional requirements of the tool will be documented in this section. Each requirement will have an ID, such as: FR-N, where N starts at 1 and increments for each Functional Requirement. Each of the Functional Requirement IDs will have a corresponding test ID listed in the RTM.

The following are the functional requirements of the Tool Runner tool.

FR-1: Invoke tool with required arguments

FR-2: Document user, computer, operating system platform, and date/time

FR-3: Verify and document the QA Status of the Tool Runner and the invoked tool

The QA Status of the Tool Runner and invoked tool is based on fulfilling the following functional requirements:

FR-4: Verify and document that tool is located in a git version-controlled repository and if so, document the code version of the Tool Runner and invoked tool.

FR-5: Determine if the local git repository branch is the “master” branch and is “clean” (i.e., files are consistent with the remote-master branch)

FR-6: Determine if the local git repository branch version is the same version as the remote “master” branch;

FR-7: Determine if Tool Runner and invoked tool are on a maintained approved tool list.

1. **Software Requirements Specifications**

The software requirements specification of the tool will be documented in this section.

Git version 2.21.0 (Windows) or 2.24.0 (Linux)

Python 3.5

Python Standard Libraries:  
logging  
pathlib  
os  
sys  
argparse  
datetime  
platform  
socket  
getpass  
subprocess

Modules:  
.\constants.py  
.\config.py (.\constants.py, pylib\config\config.py [json], pylib\autoparse\autoparse.py [argparse])  
pylib\info.info.py  
pylib\pygit\git.py

1. **Software Design Description**

The software design description of the tool will be documented in this section. The results of a Code Walkthrough with an independent third party will be summarized in this section.

The following is a brief description of the required argumentsand the ouput generated by the Tool Runner tool.

* Positional Arguments:
  + Name: the filename of tool to be invoke (if tool is an executable, tool\_filename.exe) or the command (perl, python, java) required to execute the tool
  + Arguments: the filename of the tool to be invoked (if a python or perl script or JAVA program) and the tool-specific arguments to be passed to the tool (see documentation associated with each tool)
* Optional Arguments:
  + -h, --help show help message
  + - -loglevel {I,D} verbosity of log: (I)nfo, (D)ebug; default=I
  + - -logfile LOGFILE path to a log file (default is stdout)
  + - - logfilemode {a,w} Log file mode: (a)ppend or (w)rite; default=w
* Shell file configuration:

python [directory path]/pylib/runner/runner.py [optional arguments—see above] Name Arguments

* Output: The following information is logged to the screen (default, i.e. stdout) or to a path and file, which is specified by user and is passed to the Tool Runner as an argument (- -logfile LOGFILE)
* DateTime stamp of when Tool Runner is invoke and where information is being logged
* Invoked command and arguments
* Code Version of Tool Runner
* Code Version of invoked tool
* QA Status of Tool Runner
* QA Status of invoked tool
* Username and the computer and platform used to invoke the Tool Runner and invoked tool

1. **Requirements Traceability Matrix**

A requirements traceability matrix for the tool will be documented in this section. At a minimum, the matrix will include IDs of: Functional Requirements and the corresponding Acceptance Test, along with an indication of the test result (Pass/Fail).

The requirements traceability matrix for the Tool Runner is presented in Table 1.

| **Table 1. Tool Runner Tool Requirements Traceability Matrix** | | |
| --- | --- | --- |
| **Functional Requirement ID** | **Acceptance Test ID** | **Test Case** |
| QA Level  FR-1, FR-2 | IT-1 | Installation Test |
| FR-3 FR-4 FR-5  FR-6 FR-7 | ATC-1 | Invoked tool is QA-approved |
| FR-3 FR-4 | ATC-2 | Tool Runner and invoked tool is not located in a GIT repository |
| FR-3 FR-5  FR-7 | ATC-3 | Local repository has been modified (not consistent with remote repository) |
| FR-3 FR-6  FR-7 | ATC-4 | Local git repository branch version is the same version as the remote “master” branch; |

1. **Test Plan and Cases**

The test plan for the tool will be documented in this section. Each test will have a unique ID and criteria for determining if the test result is pass or fail. The TEST ID will be referenced in the RTM and ATR. An installation test, labeled **IT-1**, will be used by the Tool Runner to confirm the version of the tool being used is running correctly before launching it with the user’s parameters.

The Unit Testing done on the tool will be documented here, also.

The test plan for the Tool Runner tool is presented in Table 2. The steps are executed in a linux command line unless otherwise specified.

| **Table 2. Tool Runner Test Plan** | | |
| --- | --- | --- |
| **TEST ID** | **Test Case** | **Test Result  (Pass/Fail)** |
| IT-1 | *Invoke Tool Runner and test tool using runner\_ITC-1.sh* | |
| Verify Tool Runner is invoked and executes |  |
| Verify invoked tool executes |  |
| ATC-1 | *Open git bash window in TESTING\_CACIE Repository* | |
| *Ensure git repository is on “master” branch; if not enter the following command:*   * *git checkout master*   *Enter the following command:*   * *git pull* | |
| *Invoke Tool Runner and test tool using runner\_ATC-1.sh* | |
| Verify that the runner\_ATC-1\_logfile.txt documents user, computer, operating system platform, and date/time |  |
| Verify that runner\_ATC-1\_logfile.txt documents the code version of the Tool Runner and test tool |  |
| Verify that Tool Runner QA Status is TEST  NOTE: Tool Runner is not on the approved tool list in the test repository—see \Test\_Repo\_Name\pylib\runner\config.json to verify |  |
| Verify that runner\_ATC-1\_logfile.txt documents that the invoked tool QA Status is QUALIFIED  NOTE: Invoked tool is on the approved tool list in the test repository—see \Test\_Repo\_Name\pylib\runner\config.json to verify |  |
| ATC-2 | *Invoke Tool Runner and test tool using runner\_ATC-2.sh* | |
| Verify that runner\_ATC-2\_logfile.txt documents that the Tool Runner and test tool are not located in a Git Repository ( ../tools/ subdirectory) |  |
| Verify that runner\_ATC-2\_logfile.txt documents that the Tool Runner and test tool QA Status is TEST |  |
| ATC-3 | *Navigate to the Test\_Repo\_Name repository and add a .txt file to directory* | |
| *Invoke Tool Runner and test tool using runner\_ATC-3.sh* | |
| Verify that runner\_ATC-3\_logfile.txt documents that the Tool Runner and test tool QA Status is TEST |  |
| *Delete addfile.txt file from directory* | |
| *Invoke Tool Runner and test tool using runner\_ATC-3.sh* | |
| Verify that Tool Runner QA Status is TEST  NOTE: Tool Runner is not on the approved tool list in the test repository—see \Test\_Repo\_Name\pylib\runner\config.json to verify |  |
| Verify that runner\_ATC-3\_logfile.txt documents that the invoked tool QA Status is QUALIFIED  NOTE: Invoked tool is on the approved tool list in the test repository—see \Test\_Repo\_Name\pylib\runner\config.json to verify | |
| *Open git bash window in Test\_Repo\_Name Repository* | |
| *Enter the following commands:*   * *git checkout development* | |
| *Invoke Tool Runner and test tool using runner\_ATC-3.sh* | |
| Verify that runner\_ATC-3\_logfile.txt documents that the Tool Runner and test tool QA Status is TEST |  |
| *Open git bash window in Test\_Repo\_Name Repository* | |
| *Enter the following commands:*   * *git checkout master* | |
| *ATC-4* | *Navigate to the Test\_Repo\_Name repository and add file addfile.txt to directory* | |
| *Open git bash window in Test\_Repo\_Name Repository* | |
| *Enter the following commands:*   * *git add addfile.txt* * *git commit ‘test commit’* | |
| *Invoke Tool Runner and test tool using runner\_ATC-4.sh* | |
| Verify that the Code Versions indicate that that local and remote repositories are not synced |  |
| Verify that runner\_ATC-4\_logfile.txt documents that the toolrunner and test tool QA Status is TEST |  |
| *Enter the following commands:*   * *git log -2* * *git reset –hard ‘first 6 characters of second commit SHA-1 hash’* | |
| *Invoke Tool Runner and test tool using runner\_ATC-4.sh* | |
| Verify that the local repository and remote repository versions are the same (no “not synced message”) |  |
| Verify that Tool Runner QA Status is TEST  NOTE: Tool Runner is not on the approved tool list in the test repository—see \Test\_Repo\_Name\pylib\runner\config.json to verify |  |
| Verify that runner\_ATC-1\_logfile.txt documents that the invoked tool QA Status is QUALIFIED  NOTE: Invoked tool is on the approved tool list in the test repository—see \Test\_Repo\_Name\pylib\runner\config.json to verify |  |



See appendix for the corresponding acceptance test case test logs as noted in the test case descriptions.

1. **Acceptance Test Report**

The test report will state whether the tool is qualified for use, summarize test case results, and report all resolved incidents and resolution of unresolved incidents.

1. **User Guide**

A guide for using the tool will be documented in this section.

The Tool Runner will be invoked using a customized shell script for each invoked tool. The shell script will include the arguments for the Tool Runner as specified in Section 4 (Software Design) and the arguments for the invoked tool.