Arithmetic-TestHarness Component

Introduction

Arithmetic testharness component is used to test the new releases of SmartFrog system in a distributed environment using an earlier stable version of SmartFrog for setting up the testharness infrastructure. It is an end-to-end solution for testing since it starts the daemons, delpoys the arithmetic example across various nodes, compares the results with the precalculated values using beanshell, generates the result as pass or failed and then stops it.

It sets up the testharness infrastructure i.e. the SmartFrog daemons with different configuration settings (security on/off, and dynamic loading on/off) before running the example across all the nodes. The example that is deployed on this infrastructure starts SmartFrog daemons of new release at different port and then deploys an arithmetic application on these daemons. Hence testing the SmartFrog system in a distributed fashion. The arithmetic application uses a beanshell component that calculates the values of the arithmetic operation to compare with the generated output of these arithmetic operations across various nodes.

How to setup Manually

1. Have a collection of machines for the setup. This may include both Linux and windows
2. Out of these choose one named as driver machine that will exports via NFS/Samba a " global file system" for all other nodes.
3. Place the stable version of SmartFrog in the shared folder something like

   /home/sf/testharness/testbed.
4. Start the SmartFrog daemons at port 3800 manually in all the nodes.
5. Generate a release zip file for the release that you want to test.
6. Copy the releases zip file in a new folder say "daemons" under the shared folder

   /home/sf/testharness.
7. Modify the

   core\components\arithmetic-testharness\src\org\smartfrog\tools\testharness\templategen\hosts.all file to add the addresses and OS type of all the nodes. For eg:

   ```
   11.14.15.105 demo-linux-7.hpl.hp.com  demo-linux-7 linux
   13.14.15.103 demo-linux-6.hpl.hp.com  demo-linux-6 linux
   11.14.15.113 demo-linux-9.hpl.hp.com  demo-linux-9 linux
   18.44.103.188 win-demo-5.hpl.hp.com win-demo-5   windows
   ```
8. Rlogin to the driver machine as user sf.
9. Run

   /home/sf/testharness/testbed/dist/bin/setupRelease <zip-release-file> demoCA demo-linux-6 demo-linux-7 demo-linux-9 win-demo-5. This script unpacks the release for "demoCA", generates CA keys, signs all the jars, create credentials for each node

   (demo-linux-6 demo-linux-7 demo-linux-9 win-demo-5) and creates a subdir under

   /home/sf/testharness/daemons for each node with its own unpacked release directory and credentials (e.g., under /home/sf/testharness/daemons/demo-linux-6/private/local).

10. Modify the file core\components\arithmetic-testharness\src\org\smartfrog\tools\testharness\templategen\templateSerrano.vm to change the environment variables like PATH, JAVA_HOME etc. in attriibute “allEnvProperties”.
11. Modify the core\components\arithmetic-testharness\src\org\smartfrog\regtest\arithmetic\templategen\hostTemplate.sf file to have the hostnames of the available nodes.
12. Run doAll in core\components\arithmetic-testharness\src\org\smartfrog\tools\testharness\templategen folder. This will generate the description files for deployment.
13. Build the arithmetic-testharness by running “ant” in arithmetic-testharness folder. Copy the jar file for the same to dist directory in /home/sf/testharness/testbed and in /home/sf/testharness/daemons/<hostname>. Copy the jar file bsh-1.3.0.jar also.
14. Deploy the "standard" test harness example in demo-linux-6 by running:

    sfStart demo-linux-6 sfConfig org/smartfrog/regtest/arithmetic/templategen/exampleTH.sf

    This should start SmartFrog daemons of the new release in all the nodes at port 3801 and their output redirected to /home/sf/testharness/daemons/<whatever>/daemon.out. After the daemons are started the arithmetic example1TH.sf is deployed with the host bindings from hostTemplate1.sf
15. Various arithmetic math examples (example1TH.sf, example2TH.sf, example3TH.sf and example4TH.sf) and host bindings (hostTemplate1.sf, hostTemplate2.sf) are available in core\components\arithmetic-testharness\src\org\smartfrog\regtest\arithmetic\templategen directory. hostTemplate1.sf has a cyclic mapping of nodes and "maximizes" the number of remote "parents". On the other hand hostTemplate2.sf does a block distribution with much better clustering. The exampleTH_Big.sf contains combinations of all these.
16. To generate random math examples (like the one example1TH.sf) execute the script

    org/smartfrog/regtest/arithmetic/templategen/doAll you will get 4 (2 small 2 big) and all the extra files you need to combine with hostTemplate1.sf and hostTemplate2.sf. Inside the script we are just calling the program:
    java org.smartfrog.regtest.arithmetic.templategen.ExampleGen -d <depletionfactor from 0-1.0 where closer to 1 means more nodes=0.9> -o <output file=example1TH.sf> -t <inputtemplate example.vm> or if you have an expression that you want to generate you could do:
    java org.smartfrog.regtest.arithmetic.templategen.ExampleGen -t <inputtemplate example.vm> -o <expresion terminated with T :"(1+G)T">
    The template (example.vm) is also a velocity template that just expands the tree, links the right outputs and fixes the generators.

Steps for Automated Testharness

1. Have a collection of Linux machines for the setup.
2. Out of these choose one named as driver machine that will exports via NFS/Samba a “ global file system” for all other nodes.
3. Place the stable version of SmartFrog in the shared folder something like /home/sf/testharness/testbed.
4. Modify the following configuration files:(ONE TIME CONFIGURATION for the setup)

   1. core/components/arithmetic-testharness/executeTest.properties file.
   2. core\components\arithmetic-testharness\src\org\smartfrog\tools\testharness\templategen\hosts.all file to add the addresses and OS type of all the nodes.
   3. core\components\arithmetic-testharness\src\org\smartfrog\tools\testharness\templategen\Remotehosts.all file to add the add user/passwd for all the nodes.
   4. core\components\arithmetic-testharness\src\org\smartfrog\regtest\arithmetic\templategen\hostTemplate.sf file to have the hostnames of the available nodes.
   5. core\components\arithmetic-testharness\src\org\smartfrog\tools\testharness\templategen\templateSerrano.vm to change the environment variables like PATH, JAVA_HOME etc. in attriibute “allEnvProperties”.

5. Build Core & Extras/ant
6. Place the release file (Zip file which needs to be tested) in core/components/arithmetic-testharness folder.
7. Run ant -f execute.xml test