- SPLAnE is available for Ubuntu/Fedora/Mac OS.
- Download Link: http://www.cse.iitb.ac.in/~splane/splane/shell/SPLAnE_Shell.zip

Installation:

1. SPLAnE require java 1.6 or later version to be installed on the PC.
2. Unzip SPLAnE_Shell.zip to any directory, and it's ready to use.
3. If you get any permission denied error, while executing any of the analysis operations, change the permission of CriQit, boole2qpro and boole2qpro.sh file. Example, chmod 777 CirQit boole2qpro boole2qpro.sh

SPLAnE Tutorial:

- You can download the realistic SPL models from: http://www.cse.iitb.ac.in/~splane/splane/models/models.zip

or the randomly generated SPL models from:

http://www.cse.iitb.ac.in/~splane/splane/examples/examples.zip

For this tutorial, we will be explaining SPLAnE on Entry Control Product Line.

- Open terminal window, and navigate to the unzip folder(SPLAnE folder).
- Type following on command line:

  $java –jar splane.jar

  Ex.

```
[gnarwane@localhost SPLAnE]$ java -jar splane.jar
```

  This will open the SPLAnE shell, as show below:

  $splane>

- Type, to display all the available commands.

  $splane>?list

  The screen shot display the available command:

```
splane> ?help
This is Cliche shell (http://cliche.sourceforg
To list all available commands enter ?list or
command-name
splane> ?list
abbrev  name    params
i       intersection    (numbers...)
v       voidmodel       ()
vs      valid-specification     (numbers...)
vs      valid-specification     ()
vi      valid-implementation    ()
vi      valid-implementation    (numbers...)
c       common  (p1)
l       live    (p1)
d       dead    (p1)
c       critical        (p1, p2)
u       union   (numbers...)
ct      complete-traceability   ()
i       implement       ()
im      implement       (numbers...)
r       realizes        (numbers...)
co      covers  (numbers...)
c       completness     ()
s       soundness       ()
vfm     valid-feature-model     ()
vcm     valid-component-model   ()
vm      valid-model     ()
lfm     load-feature-model      (p1)
lcm     load-componente-model   (p1)
lf      loadtrace-file  (p1)
splane>
```

- You can either use a command name or abbreviation.

- You can check the syntax of each command as below:

```
splane> ?help covers
Command: covers
Abbrev:  co
Params:  (numbers...)
Description: covers(numbers:String[]) : String
Number of parameters: 1
numbers String[]        add the implications, a comma and the specifications
This command is varargs on its last parameter.
```

- Now, load the SPL model (feature model, component model, traceability model) for Entry Control Product Line as shown below: (This is required before running any other analysis operations).

```
splane> lfm ./examples/fEntryControl.afm
Feature Model loaded successfully.

splane> lcm ./examples/cEntryControl.afm
Component Model loaded successfully.

splane> lf ./examples/tEntryControl.afm
Traceability Model loaded successfully.

splane>
```

- On successfully loading the model, check is traceability is complete or not. Traceability should be complete, or other operations may fail.

```
splane> ct
Execution Time:0
Traceability Relation is Complete
.

splane> complete-traceability
Execution Time:0
Traceability Relation is Complete
.
```

- Check if, feature model and component model is valid or not. Type:

```
splane> vfm
Execution Time(ms):6
The Feature Model is VALID.

splane> vcm
Execution Time(ms):6
The Component Model is VALID.
```

- Check if, SPL can generate any product or not (as below):

```
splane> v
Execution Time(ms):9
The SPL Model is NOT VOID.
```

- Check for valid Specification:

```
splane> vs fEntryControl fPowerLock fControlMode fAutomatic
Execution Time(ms):7
Specification:[fEntryControl, fPowerLock, fControlMode, fAutomatic]
The Specification is VALID.
```

Alternate way:

```
splane> vs
Select Features in Specification(Yy/Nn):
fManualLock :n
fManual :n
fShiftOutOfPart :n
fDoorLock :n
fPowerLock :y
fSpeed :n
fAutomatic :y
fEntryControl :y
fLockMode :n
fControlMode :y
fDoorRelock :n
Execution Time(ms):6
Specification:[fPowerLock, fAutomatic, fEntryControl, fControlMode]
The Specification is VALID.
```

Ex, Invalid Specification:

```
splane> vs fEntryControl fPowerLock fControlMode fAutomatic fManualLock
Execution Time(ms):5
Specification:[fEntryControl, fPowerLock, fControlMode, fAutomatic, fManualLock]
The Specification is NOT VALID.
```

- Check for valid Implementation:

```
splane> vi cEntryControl cPowerLock cDoorLockManager cControlMode cCourtesySwitch cKeySignal cDoorSignal cAutomatic
Execution Time(ms):8
Implementation:[cEntryControl, cPowerLock, cDoorLockManager, cControlMode, cCourtesySwitch, cKeySignal, cDoorSignal, cAutomatic]
The Implementation is VALID.
```

Alternate way:

```
splane> vi
Select Components in Implementation(Yy/Nn):
cDoorSignal :y
cManualLock :n
cCourtesySwitch :y
cAutomatic :y
cManual :n
cPowerLock :y
cAutoLock :n
cControlMode :y
cSpeed :n
cDoorLockManager :y
cKeySignal :y
cEntryControl :y
cLockMode :n
cGearInPark :n
Execution Time(ms):7
Implementation:[cDoorSignal, cCourtesySwitch,
The Implementation is VALID.
```

Ex, of Invalid Implementations:

```
splane> vi cEntryControl cPowerLock cDoorLockManager cControlMode cCourtesySwitch cKeySignal cDoorSignal cAutomatic cManual
Execution Time(ms):6
Implementation:[cEntryControl, cPowerLock, cDoorLockManager, cControlMode, cCourtesySwitch, cKeySignal, cDoorSignal, cAutomatic, cManual]
The Implementation is NOT VALID.
```

- Check if, Implementation implements a feature:

```
splane> implement cEntryControl cPowerLock cDoorLockManager cControlMode cCourtesySwitch cKeySignal cDoorSignal cAutomatic , fPowerLock
Execution Time(ms):7
[cEntryControl, cPowerLock, cDoorLockManager, cControlMode, cCourtesySwitch, cKeySignal, cDoorSignal, cAutomatic] IMPLEMENTS fPowerLock
```

Alternate way:

Ex, Invalid case:

```
splane> implement
Select Components in Implementation(Yy/Nn):
cDoorSignal :y
cManualLock :n
cCourtesySwitch :y
cAutomatic :y
cManual :n
cPowerLock :y
cAutoLock :n
cControlMode :y
cSpeed :n
cDoorLockManager :y
cKeySignal :y
cEntryControl :y
cLockMode :n
cGearInPark :n
Enter Feature :
fManual
Execution Time(ms):331
[cDoorSignal, cCourtesySwitch, cAutomatic, cPowerLock, cControlMode, cDoorLockManager, cKeySignal, cEntryControl] does not IMPLEMENTS fManual
```

- Check if, the Implementation realize the Specification:

```
splane> realizes cEntryControl cPowerLock cDoorLockManager cControlMode cCourtesySwitch cKeySignal cDoorSignal cAutomatic , fEntryControl fPowerLock f
ControlMode fAutomatic
Execution Time(ms):8
[cEntryControl, cPowerLock, cDoorLockManager, cControlMode, cCourtesySwitch, cKeySignal, cDoorSignal, cAutomatic] REALIZES the specification [fEntryCo
ntrol, fPowerLock, fControlMode, fAutomatic]
```

Invalid case:

```
splane> realizes cEntryControl cPowerLock cDoorLockManager cControlMode cCourtesySwitch cKeySignal cDoorSignal cAutomatic cAutoLock cLockMode cSpeed ,
 fEntryControl fPowerLock fControlMode fAutomatic
Execution Time(ms):6
[cEntryControl, cPowerLock, cDoorLockManager, cControlMode, cCourtesySwitch, cKeySignal, cDoorSignal, cAutomatic, cAutoLock, cLockMode, cSpeed] does n
ot REALIZES the specification [fEntryControl, fPowerLock, fControlMode, fAutomatic]
splane>
```

- Check if, the Implementation covers the specifications:

```
splane> covers cEntryControl cPowerLock cDoorLockManager cControlMode cCourtesySwitch cKeySignal cDoorSignal cAutomatic cAutoLock cLockMode cSpeed , f
EntryControl fPowerLock fControlMode fAutomatic
Execution Time(ms):7
[cEntryControl, cPowerLock, cDoorLockManager, cControlMode, cCourtesySwitch, cKeySignal, cDoorSignal, cAutomatic, cAutoLock, cLockMode, cSpeed] COVERS
 the specification [fEntryControl, fPowerLock, fControlMode, fAutomatic]
```

Invalid case:

```
splane> covers cEntryControl cPowerLock cDoorLockManager cControlMode cCourtesySwitch cKeySignal cDoorSignal cAutomatic , fEntryControl fPowerLock fCo
ntrolMode fAutomatic fDoorLock fLockMode fSpeed
Execution Time(ms):11
[cEntryControl, cPowerLock, cDoorLockManager, cControlMode, cCourtesySwitch, cKeySignal, cDoorSignal, cAutomatic] does not COVERS the specification [f
EntryControl, fPowerLock, fControlMode, fAutomatic, fDoorLock, fLockMode, fSpeed]
```

- Checking Soundness and completeness property over SPL:

```
splane> soundness
Execution Time(ms):10
SPL hold SOUNDNESS Property.

splane> c
Execution Time(ms):10
SPL does not hold COMPLETNESS Property.

splane>
```

- Check if, component is critical for a given feature:

```
splane> critical fManual cManual
Execution Time(ms):26
"cManual" is CRITICAL for feature "fManual".

splane> critical fAutomatic cManual
Execution Time(ms):17
"cManual" is NOT CRITICAL for feature "fAutomatic".
```

- Check if, element is common in all the product of SPL:

```
splane> common fPowerLock
Execution Time(ms):316
"fPowerLock" is COMMON in all the products.

splane> common fManual
Execution Time(ms):20
"fManual" is not COMMON in all the products.
```

- Check if, element is present(live) in atleast one of the product:

```
splane> live cDoorLockManager
Execution Time(ms):23
"cDoorLockManager" is LIVE in SPL.

splane> live fAutomatic
Execution Time(ms):22
"fAutomatic" is LIVE in SPL.

splane> live fManualLock
Execution Time(ms):22
"fManualLock" is not LIVE in SPL.
```

- Check if, element is not present(dead) in any of the product:

```
splane> dead fManualLock
Execution Time(ms):24
"fManualLock" is DEAD in SPL.

splane> dead fDoorLock
Execution Time(ms):21
"fDoorLock" is not DEAD in SPL.
```

- Check if, the Union of two specification, Implementation or product can give any valid specification, Implementation or product respectively:

```
splane> u S fEntryControl fPowerLock fControlMode fAutomatic , fEntryControl fPowerLock fControlMode fDoorLock fLockMode fSpeed fAutomatic
Execution Time(ms):23
UNION is possible.

splane> u S fEntryControl fPowerLock fControlMode fAutomatic , fEntryControl fPowerLock fControlMode fDoorLock fLockMode fSpeed fManual
Execution Time(ms):15
UNION is not possible.
```

- Check if, the Intersection of two specification, Implementation or product can give any valid specification, Implementation or product respectively:

```
splane> intersection S fEntryControl fPowerLock fControlMode fAutomatic , fEntryControl fPowerLock fControlMode fDoorLock fLockMode fSpeed fAutomatic
Execution Time(ms):23
INTERSECTION is possible.

splane> intersection S fEntryControl fPowerLock fControlMode fAutomatic , fEntryControl fPowerLock fControlMode fDoorLock fLockMode fSpeed fManual
Execution Time(ms):23
INTERSECTION is not possible.
```