Performance testing tool user guide:

Prerequisite:

1. Should be New Exility based Project.
2. Test case preparation: User needs to enter transaction Name & start Exility test case recorder & enter all the details in the application. System automatically create one excel book for each service entry . This contains 2 types of excel sheets :
   1. Input details(Prefixed with ‘in\_’ ) which was sent from client.
   2. Output details(Prefixed with ‘out\_’) which are sent to client.
3. For performance testing only input excel sheets are used.

Preparation:

1. Identify Key generation filed: Developer needs to identify, which are the input field needs to be changed while posting transaction; those details should be marked as ‘Load Key’ generation in Excel sheet. In WeaveIT application performance testing, we used ##LoadKey## to generate new number & ##RepeatLoadKey## to reuse already generated number to subsequent record in the same grid.
2. Create Control Sheet: Before this Key generation field should be updated before creating control sheet. This option is provided in Exility generator(resource):
   1. Choose SoupUI Script generator Menu,
   2. Select ‘Control Sheet’ as a Generate option.
   3. Enter Test case path
   4. Press on ‘Create control sheet’, one consolidated control sheet excel book will be created in test case path.
3. Now developer/tester needs to specify how dynamic key needs to be generated in control sheet(which was generated using 2nd step). Choose appropriate Dynamic key generation pattern by updating correct ‘Range Classification’ field
   1. Key generation patterns are broadly classified into 5 types, this is concluded based on Range classification:
      1. Simulation: this generates sequential number with specified pattern
         * Specify Prefix (suffix) if any ex: local PO number to be prefixed with type of PO, financial year. Then prefix column should be updated in with ‘LPO/2013-2014/’.
         * Provision is made to generate fixed number length ex: in case number needs to be generated with 5 digit numbers & shortfall should be filled with 0 then Pad Length should be updated as to be generated also padding facility ex: 00013.
         * Start & end range.
         * Range Classification:
           1. STOP: it stop generating next number once it reaches to End range
           2. RESTART: it restart the generating number from start number once it reaches to end number Ex: Material code generation is specified between 1 and 1000. After generating 1000 it automatically it start generating number from 1
           3. RANDOM: Randomly generate number between start and end range. (Not tested)
           4. DATE\_ADD\_STOP : Current date will be added with Last generated Number. it stop generating Next date once it reaches to End range.
           5. DATE\_ADD\_RESTART: It initialize to Current date + start range number once it reaches to end range number.
      2. Synchronize with data context: This case should be used when dynamic key to be generated based on another value. Ex: While creating GRN number generation Material number should be generated based on PO number. In this case Material identification query should be updated in query field, replace Key should be updated as PO number & Range Classification should be updated as ‘SYNC’.
      3. EMPTY : to set Empty content. Arul asked this for a specific context. Hence it is added. This nullifies the content. To use this simply update Range classification as ‘EMPTY’. (May not be required now).
      4. CUSTOMISE : Where multiple number to be generated & existing number generation is not feasible(time consuming) for performance testing in that situation this case can be used. Here user has a complete hold on what they wanted to perform & manipulate service data. To use this range classification should be updated as ‘CUSTOMISE’ & update query field as fully qualified java class name. (Not tested)
4. Create ‘TestCaseKeyGeneration’ table & upload control sheet(which was updated using 3rd step) content in Application database.
5. Generate SAOPUI script :- Option is provided in Exility generator(resource):
   1. Choose SoupUI Script generator Menu,
   2. Select ‘*SOAP UI Script*’ as a Generate option.
   3. Enter Test case path (Generated test case , can be specified in parent folder)
   4. Enter URL ( Service execution path ex: http://localhost:8081/Evincus/Serve)
   5. Enter Project Name ( SoapUI script File will be generated with this name)
   6. Press on Get SOAPUI, Soup UI script will be generated on test case folder*.*
6. Now import SoupUI script(which was generated in 5th step) using ‘File->Import Project’ in SoapUI Application. Run this test case to verify everything is configured according to requirement.
7. Create a Project in LOADUI project & attach SOAPUI test case & save. Now system is ready for testing.

Note: Load UI Script is system dependent.