SDTF is en unit testing framework geschreven in python en gebaseerd op de python unittest library (PyUnit). Het doel van het framework is om stukken code (units) te gaan testen met verschillende inputs en kijken of de werkelijke output hetzelfde is als de verwachte output. SDTF zorgt ervoor dat dit proces automatisch verloopt en wordt teruggekoppeld naar een test management system. In ons geval is dit TestLink.

SDTF is een uitbreiding op pyunit dat zorgt voor onder andere een shared context object dat het mogelijk maakt om informatie tussen test cases en test suites door te geven. Een andere feature van SDTF is dat het mogelijk maakt om de test resultaten door te geven naar TestLink.

Ook op te merken is dat SDTF gebruikt maakt van third party software voor sommige tests. Een voorbeeld is Selenium, een suite voor browser automatisatie.

## SystemTest

Systemtest bevat de uitbreidingen op de unittest framework en is gedefinieerd in framework/core/systemtes.py.

### SystemTestCase

SystemTestCase is een subclass van de unittest.TestCase class, deze zorgt ervoor dat het log berichten kan opvangen tijdens het uitvoeren van de testcase. De log berichten worden opgenomen door een custom subclass van de logging.Handler class.

Als de test case geen succes is wordt get\_known\_issues opgeroepen. De return values hiervan worden gebruikt om TestLink te updaten met de ID’s van de issues.

### SystemTestSuite

Dit bevat een setUp en een tearDown. Dit wijkt af van pyunit, waar het normaal bij elke testcase gebeurt.

Dan is er ook nog dit: “The suites can be used to represent the directory structure of the test case organization in TestLink”, maar ik begrijp niet zo goed wat dit inhoud.

### SystemTextTestResult

Subclass van TextTestResult, voegt de geslaagde test cases toe aan het testResulat object, omdat de geslaagde test ook moeten worden geregistreerd in het Test Management System.

Alle addXXX methods (addSuccess, addError en addFailure) zijn uitgebreid zodat ze onmiddellijk test case resultaten registreren in het Test Management System. Voor TestLink is het mogelijk om de build ID automatisch aan te maken in TestLink voor het geselecteerde testplan, indien mogelijk. Maar enkel als de nieuwe build ID bevestigd is door het uitvoeren van de test case.

### SystemTestLoader

Het builden van een test suite wordt geautomatiseerd met een testLoader instance; gebaseerd op conventies omtrent naamgeving.

## Logging

Voor logging gebruiken we de standaard Python library. Tijdens het uitvoeren van een test case wordt elk bericht dat geprint wordt in de standaard output gecaptured en naar de logger gestuurd. Maar het voordeel van de logger rechtstreeks te gebruiken is dat filenaam en lijnnummer ook wordt opgenomen, zodat je weet van waar het bericht afkomstig is.

## Configuratie

De configuratie wordt geladen uit 2 bronnen, de ene is centraal in het framework, de ander is gelinkt aan het uit te voeren test project door zijn plek in het file system. De framework/core/config.py module definieert de configuratie instellingen die gemeenschappelijk zijn voor alle tests.

## Main: program execution

De code om de config te laden, een test suite te bouwen, runnen en rapporteren naar de Test Management System is geconcentreerd in de Main class gedefinieerd in framework/core/main.py. Deze class kan worden gebruikt zonder parameters aangezien er logische defaults worden voorzien.

## STAF

Kort voor Software Testing Automation Framework. Dit is een multi-platform multi-language framework dat hier gebruikt wordt om processen remote op te starten en files van en naar een remote host te kopiëren.