CLEARSY
Safety Solutions Designer

NOV 2021

AIX
LYON
PARIS
STRASBOURG

WWW.CLEARSY.COM

Logiciel de test automatique

CLEARSY Test Framework

CONTACT@CLEARSY.COM

Sommaire

GENERALITES

FONCTIONNEMENT

NOTRE OFFRE

CLEARSY Test Framework

CLEARSY Test Framework est un atelier logiciel destiné à réaliser les **tests fonctionnels boîte noire** d'un **système**.

Principes généraux

Un test stimule les interfaces en entrée et vérifie les sorties**.**

Des conditions complexes peuvent être définies pour la vérification.

Un ensemble de tests constitue un scénario.

Les scénarios sont rassemblés dans des fichiers XML ouverts à l'utilisateur.

L'exécution séquencée d'un ensemble de scénarios peut être réalisée pour des tests de non-régression.

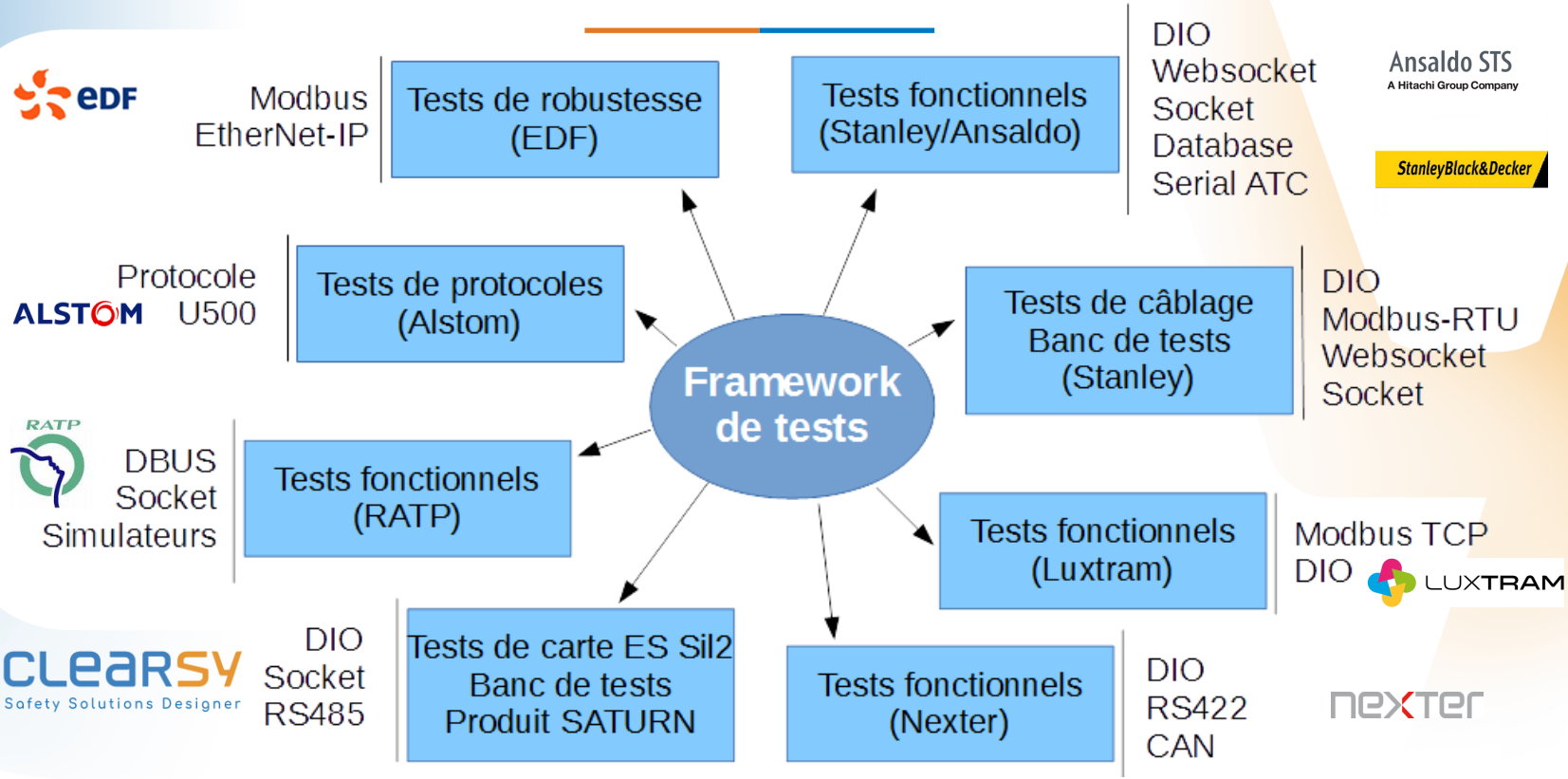
Caractéristiques

- Adapté au domaine industriel :
  Qualifié T2 CEI 61508 et EN50128 (pour la solution générique)
- Evolutif : adaptable à toute spécificité concernant les tests
- Exécution sur un PC standard
- Windows et Linux
- Standards XML et Xunit
- Génération de rapports formatés (docx)

- Développé depuis 2014
- Utilisé sur plus de 20 projets

CLEARSY Safety Solutions Designer — RATP — ALSTOM — LUXTRAM — StanleyBlack&Decker — nexter — EDF

Domaines d'application : Interfaces - Références



Sommaire

GENERALITES

FONCTIONNEMENT

NOTRE OFFRE

Organisation d’un test

Un test obéit au principe suivant :

- Etant donné un contexte (situation de départ)
- Lorsque le Framework de tests effectue certaines actions sur une ou des interfaces
- Alors il doit pouvoir constater certaines conséquences sur une ou des interfaces

Constituants

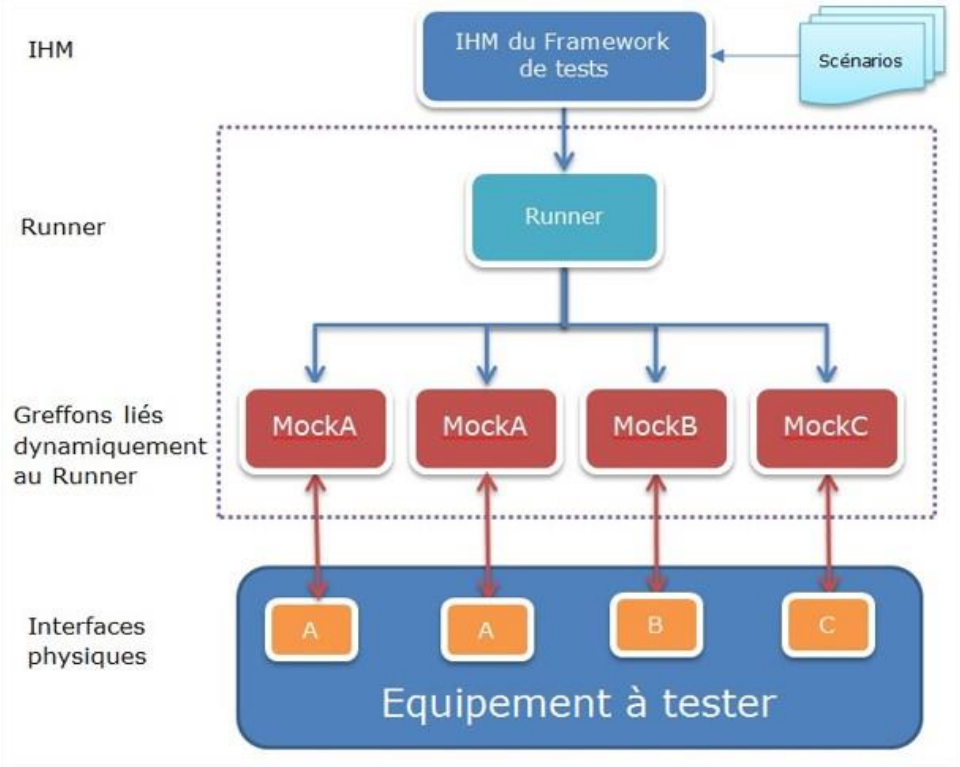
Le Framework de tests comprend :

- Une **interface graphique** qui permet de lancer des tests et de présenter le résultat
- Le **Runner** (c'est le séquenceur, le cœur), disponible sous la forme d'un logiciel exécutable
- Les **Mocks**, modules qui étendent les fonctionnalités du Runner
  - Un Mock implémente typiquement le comportement d'un protocole
  - Les Mocks sont instanciés et pilotés par le Runner
  - Un Mock est à prévoir par interface à tester (ou plusieurs instances d'un même Mock si l'interface est multiple)

Environnement : C++ Qt

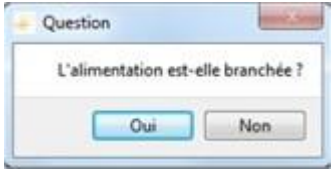
Multi plateforme : **Windows** ou **Linux**

Architecture

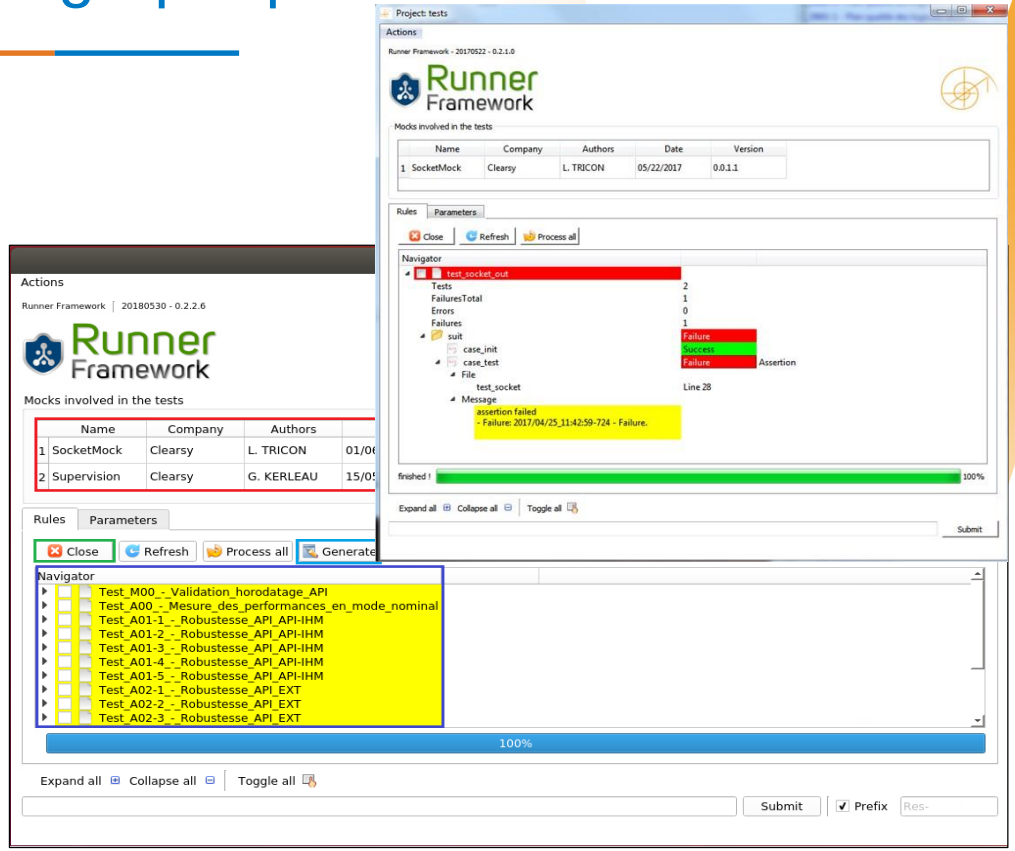


Interface graphique

- Charge les scénarios
- Lance les tests
- Affiche des pop-ups d'interaction utilisateur requises par les tests



- Génère les rapports de test



Exemple de test (fichier XML)

Programmation d'un test par l'utilisateur (via un Editeur XML standard).

Ce test utilise deux mocks « TCPModbus » qui mettent a disposition des commandes et des remontées d'information sur la liaison Modbus (***set*** et ***timercompare***).

mocks à instancier

```
<test_runner name="Test de la fonction Gestion d'aiguille motorisée">
  <tools>
      <tool id="modbus_CMD_API"   path="TcpModbusMock_API_CMD.dll"   init="mapping-API-CMD.xml"/>
      <tool id="modbus_INFOS_API" path="TcpModbusMock_API_INFOS.dll" init="mapping-API-INFOS.xml"/>
  </tools>
  <suits>
    <suit name="Fonctionnement de la sortie Cmmag" txt="Cette suite vérifie les filtrages de la sortie Cmmag">
        <case name="Filtrage de la sortie CmmAg à l'activation">
            <action tool="modbus_CMD_API"   exec="set E_Cmmag 1"/>
            <action tool="modbus_INFOS_API" exec="timercompare S_CmmAg 2000 0 1"/>
        </case>
        <case name="Filtrage de la sortie CmmAg à la désactivation">
            <action tool="modbus_CMD_API"   exec="set E_Cmmag 0"/>
            <action tool="modbus_INFOS_API" exec="timercompare S_CmmAg 500 1 0"/>
        </case>
    </suit>
  </suits>
</test_runner>
```

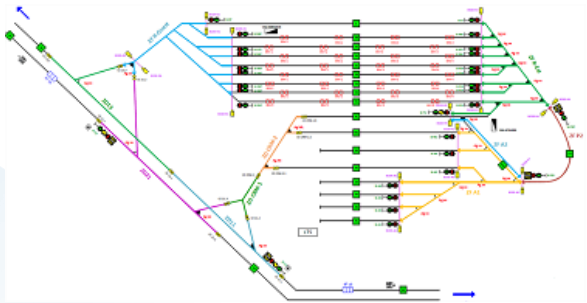
Cas de test
chaque action retourne vrai ou faux

set	met à jour la variable E_CmmAg avec la valeur 0
timercompare	vérifie que la valeur de S_CmmAg change après 500 ms de la valeur 1 à la valeur 0

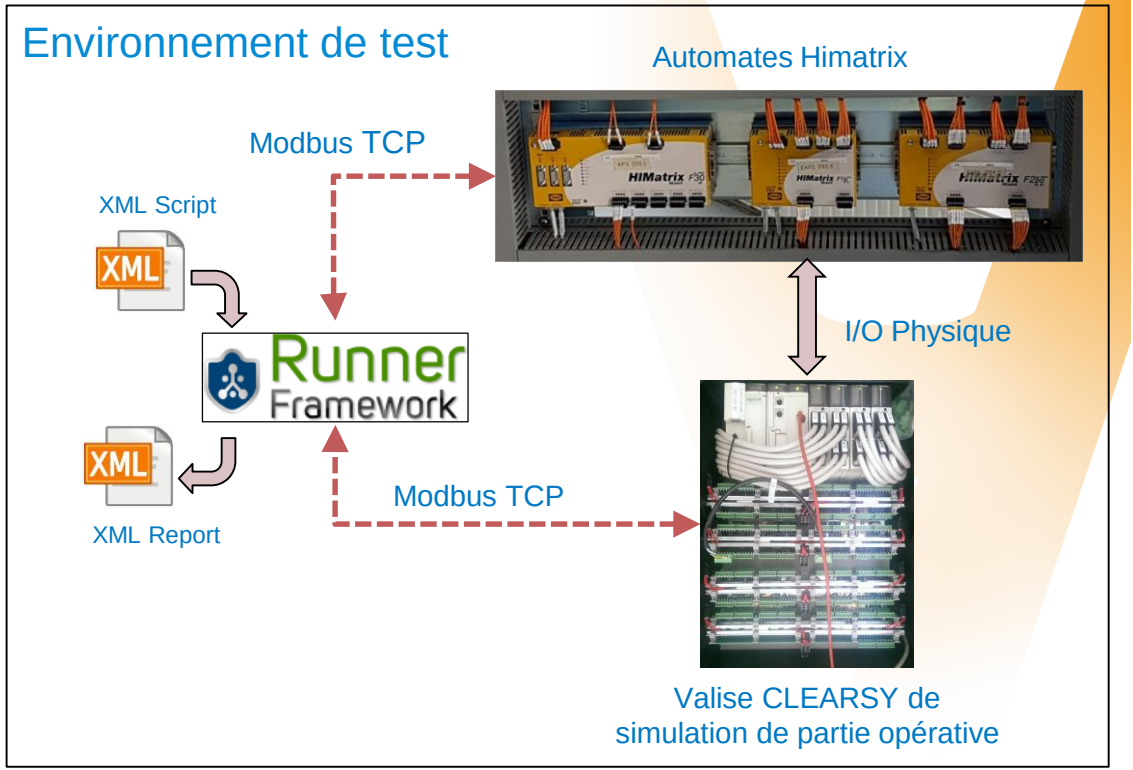
Cas d'utilisation : Poste de manœuvre SIL4

Test de logiciel automate de contrôle-commande de signalisation

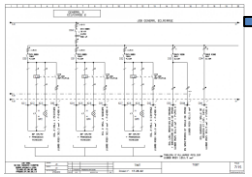
Simulation d'environnement (feux, fins de course aiguilles, circuits de voie…) et ensemble des Tests fonctionnels compatibles avec EN50128:SIL4



Environnement de test



Cas d'utilisation : Test d'armoires électriques



Electrical diagrams

XML script
Refers to:
Mock_oscillo
Mock_doortalk
LUA script(s)

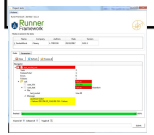
Test Framework

Shared variables
- Points array
- OK/KO flag

Runner (Core)

Mock_oscillo

Mock_doortalk



ActiveX
Oscillo
Server

Test platform

LUA script
Points treatment (max, min, avg..)
OK/KO flag set

LUA script: Computation library developped by Stanley and called by XML scripts (use of shared variables)

XML Report

Points (csv)

Test Report

StanleyBlack&Decker

RS232
doortalk protocol

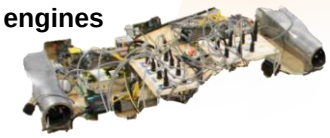
MC251
Engine controller

Eth

Oscillo

Electrical cabinet

Doors engines



Banc de test de contrôle-commande de porte palière

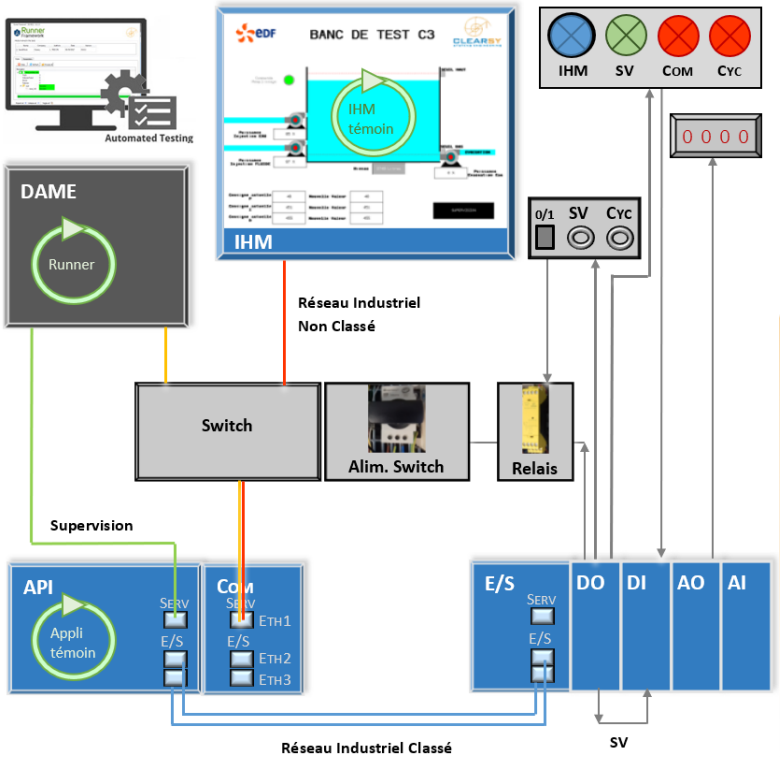
Vérification automatisée de câblage d'armoires de contrôle-commande pour le métro d'Honolulu : tests de conformité aux schémas électriques

Cas d'utilisation : Qualification de communications C3 Cybersécurité

Qualification de la robustesse d'automate industriel (API) classé
vis-à-vis de pollutions provenant de systèmes non classés (IHM)

Utilisable pour des tests de cybersécurité (déni de service)

Valise de tests de qualification



Sommaire

GENERALITES

FONCTIONNEMENT

NOTRE OFFRE

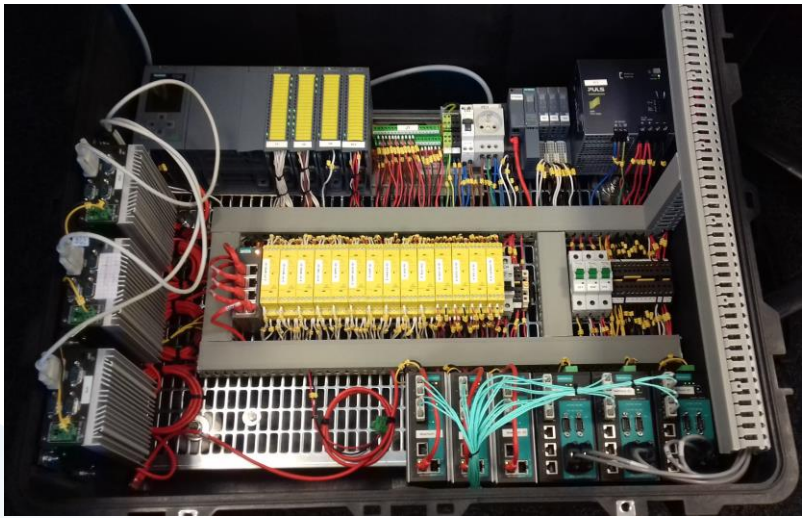
Notre offre

- La Solution générique (plug and play) comprend
  - Les protocoles disponibles : Modbus/TCP, Modbus/RTU, CAN
  - Des conditions de test standards :
    Opérations = < > sur valeurs booléennes, entiers ou chaines
- Nous développons votre solution spécifique
  - Ajout des protocoles nécessaires : interfaces et couches applicatives
  - Ajout de conditions de test
  - Mise en forme sur mesure des rapports (format docx)
- Services proposés
  - Co-définition des besoins
  - Développement de la solution spécifique
    Livraison des exécutables et Manuel d'Installation et Utilisation (inclut langage de test et exemples)
  - Développement et livraison du banc complet avec le matériel
    ou Support au développement du banc
  - Développement des tests *ou* Support au développement des tests

Exemples de bancs de test réalisés par CLEARSY

Bancs de test réalisés sur mesure

à partir de spécifications techniques fournies par le client ou définies par CLEARSY



Banc de test pour RATP

Banc de test pour EDF

Avantages

- Pas de licence d'utilisation si CLEARSY livre le banc complet
- Un banc de test sur mesure qui part des besoins client
  et non pas une adaptation des besoins à un outil existant
  - adaptation sur mesure des interfaces, des protocoles, des tests, des rapports
- Autonomie du client sur l'ajout/modification de tests
- Possibilité de réutilisation du banc sur tout projet
- Un seul interlocuteur ayant les compétences en logiciel et matériel
- Time-to-market avantageux
  - durée moyenne d'un projet : de 3 à 6 mois

Contacts

contact@clearsy.com

www.clearsy.com

https://www.clearsy.com/outils/testrunner/