RE000 Requirements for the Library codecs_lib

Conventions

Requirements listed in this document are constructed according to the following structure:

Requirement ID: REQ-UVW-XYZ

Title: Title / name of the requirement

Description: Description / definition of the requirement

Verification Method: I / A / T / D

The requirement ID starts with the fixed prefix 'REQ'. The prefix is followed by 3 letters abbreviation (in here 'UVW'), which defines the requiement type - e.g. 'FUN' for a functional and capability requirement, 'AWM' for an alarm, warnings and operator messages, etc. The last part of the ID is a 3-digits *hexadecimal* number (0..9|A..F), with the first digit identifing the module, the second digit identifing a class / function, and the last digit - the requirement ordering number for this object. E.g. 'REQ-FUN-112'. Each requirement type has its own counter, thus 'REQ-FUN-112' and 'REQ-AWN-112' requirements are different entities, but they refer to the same object (class or function) within the same module.

The verification method for a requirement is given by a single letter according to the table below:

Term	Definition
Inspection (I)	Control or visual verification
Analysis (A)	Verification based upon analytical evidences
Test (T)	Verification of quantitative characteristics with quantitative measurement
Demonstration (D)	Verification of operational characteristics without quantitative measurement

Functional and capability requirements

Requirement ID: REQ-FUN-000

Title: COBS codec

Description: The library should provide an encoder / decoder based on the COBS algorithm.

Verification Method: A

Requirement ID: REQ-FUN-001

Title: Vigenere cipher codec

Description: The library should provide an encoder / decoder based on the Vigenere cipher using an externally provided key.

Verification Method: A

Requirement ID: REQ-FUN-002

Title: WH random generator codec

Description: The library should provide an encoder / decoder based on the Wichmann and Hill pseudo-random numbers generator using an externally provided key as the seed.

Verification Method: A

Requirement ID: REQ-FUN-003

Title: Simple, no-key scrambler

Description: The library should provide an encoder / decoder based on a reversible data scrambling without a key.

Verification Method: A

Interfaces

Requirement ID: REQ-INT-000

Title: Reliable dependencies

Description: Except for the packages developed and maintained at Diagnoptics the library should be based either solely on the Standard Python Library, or it should use only widely accepted / used and well maintained 3rd party libraries / packages.

Verification Method: I

Installation and acceptance requirements

Requirement ID: REQ-IAR-000

Title: Python interpreter version

Description: The library should be used with Python 3 interpreter. The minimum version requirement is Python v3.6.

Verification Method: D

Requirement ID: REQ-IAR-001

Title: Operational system

Description: The library should work, at least, under MS Windows and GNU Linux operational systems. Ideally, it should not utilize any platform-specific functionality, therefore it should work under any OS, for which Python 3 interpreter is available.

Verification Method: D

Requirement ID: REQ-IAR-002

Title: System requirements check

Description: The library should provide a module / script to check if all system requirements are met, i.e. the Python interpreter version, other required libraries / packages presence as well as their versions. This module / script should report the missing requriements.

Verification Method: D

User documentation requirements

Requirement ID: REQ-UDR-000

Title: The library is thoroughly documented.

Description: The library should be documented, including:

- Requirements documents
- Test reports
- User and API references

The reference documentation should provide sufficient data on the implementation for the future maintenance and modification as well as clear and comprehensive usage instructions and examples.

Verification Method: I