

Qualitätssicherungsmaßnahmen

Gebrauchsanleitung:

Für den sicheren und bestimmungsgemäßen Umgang mit der Maschine haben wir eine Gebrauchsanweisung verfasst. Diese ist unter *“doku/Betriebsanleitung”* zu finden.

Programmierstil:

Wir erstellen Quellcode nach bestimmten, vorab festgelegten Regeln. Dies ist insbesondere für die Verständlichkeit und Wartbarkeit unserer Software wichtig. Der festgelegte Programmierstil ist unter *“doku/CodingStyle”* zu finden.

Unit Test:

Um den konkreten Code zu testen, benutzen wir Unit-Tests. Dafür wurde ein Framework erstellt *“ESEP/src/test/unit”*. Die Unit-Tests sind unter *“ESEP/src/test/ut”* zu finden.

```
Running unit test suites: .....
    Analyser::Expand: 100% [ 3 / 3] tests successful
      BDSCAN: 100% [ 4 / 4] tests successful
    ByteStream: 100% [ 6 / 6] tests successful
    CRC32 Generator: 100% [ 5 / 5] tests successful
    Communication Layer: 100% [ 6 / 6] tests successful
      Compound Enum: 100% [ 5 / 5] tests successful
    Configuration Object: 100% [ 5 / 5] tests successful
      EMP Parser: 100% [ 3 / 3] tests successful
    EMP Playback HAL: 100% [ 4 / 4] tests successful
      EMP Tokenizer: 100% [ 4 / 4] tests successful
    Error Manager Logic: 100% [15 / 15] tests successful
    Finite State Machine: 100% [ 5 / 5] tests successful
      Functor Chain: 100% [ 2 / 2] tests successful
      HW Location: 100% [ 5 / 5] tests successful
      Master Logic: 100% [20 / 20] tests successful
      Overflow Buffer: 100% [ 6 / 6] tests successful
      Process Tree: 100% [ 6 / 6] tests successful
      Processor: 100% [ 3 / 3] tests successful
      QNX Connections: 100% [ 2 / 2] tests successful
    Run Manager - Time Controller: 100% [ 9 / 9] tests successful
      Serial Client (BSP): 100% [ 9 / 9] tests successful
      Serial Connection (Dummy): 100% [ 5 / 5] tests successful
        Stream: 100% [ 6 / 6] tests successful
      Test HAL: 100% [ 8 / 8] tests successful
      Timer: 100% [12 / 12] tests successful
      Watchdog: 100% [ 4 / 4] tests successful
    log/format::parser: 100% [ 6 / 6] tests successful
      sync/Container: 100% [ 5 / 5] tests successful

SUCCESS!
```

Code Qualität:

Wenn zuviel Code in einer Quelltextdatei oder einer Funktion ist, wird es schnell unübersichtlich. Deshalb lagern wir in so einem Fall den Code auf mehrere Dateien bzw. Hilfsfunktionen aus. Dies dient in erster Linie dazu die Leserlichkeit und die Wartbarkeit zu verbessern. Dies ist z.B. in unserem RunManager-Modul zu sehen. Hier wurde der Code für die Initialisierung in eine Hilfsfunktion ausgelagert *“ESEP/src/base/run/init_Logic.cpp”*.

Funktionale Tests:

Wir überprüfen unser System in Bezug auf funktionale Anforderungsmerkmale, also ob das Verhalten, der Software bzw. des Systems den Anforderungen(Requirements) im Pflichtenheft entspricht. Ein Beispiel dafür ist der funktionale Test des Höhensensors. Dieser und andere Funktionstests sind in *"Esep/src/test/ft"* zu finden.

Akzeptanztest:

Wir überprüfen ob unsere Software die funktionalen Erwartungen und Anforderungen im Gebrauch erfüllt. Dies ist z.B. an dem Run-Manager-Test unter *"Esep/src/test/ft/runmanager"* zu sehen.

Dokumentation:

Die mit dem Kunden in den Meetings festgelegten Beschlüsse sind unter *"doku/Spezifikationsbeschluesse mit dem Kunden"* zu finden. Die Beschlüsse, die im Team getroffen worden sind, sind unter *"doku/beschluesse"* zu finden.

Modellierung & Design:

Modellierung unserer Software ist unter *"doku/diagrams"* zu finden. Diese dienen z.B. dazu bei Unklarheiten sich wieder einen Überblick verschaffen zu können.