**Disposition zur Masterarbeit**

ETIC2 zur Verwaltung von Ventiltests

Student: Andreas Stucki

Betreuer: Marc-André Bumann

Datum: 25.02.2017

Inhaltsverzeichnis

[1 Projektbeschreibung / Ausgangslage 3](#_Toc475890582)

[1.1 Situation 3](#_Toc475890583)

[1.2 Problemstellung 3](#_Toc475890584)

[1.3 Ist-Zustand 3](#_Toc475890585)

[1.3.1 Stärken 4](#_Toc475890586)

[1.3.2 Schwächen 4](#_Toc475890587)

[2 Zielsetzung 4](#_Toc475890588)

[2.1 Quantitative Ziele 5](#_Toc475890589)

[2.2 Qualitative Ziele 5](#_Toc475890590)

[3 Aufgabenabgrenzung 5](#_Toc475890591)

[4 Vorgehenssystematik 5](#_Toc475890592)

[4.1 Wirtschaftliche Aspekte 6](#_Toc475890593)

[4.2 Zeitplan 6](#_Toc475890594)

[5 (erwartete) Resultate 6](#_Toc475890595)

[6 Glossar 7](#_Toc475890596)

[7 Abbildungsverzeichnis und Tabellenverzeichnis 7](#_Toc475890597)

[8 Literaturverzeichnis 7](#_Toc475890598)

# Projektbeschreibung / Ausgangslage

## Situation

Die Firma VAT stellt Vakuumventile her für Halbleiter- und Medizinalindustrie, Forschung und Entwicklung sowie Automobilindustrie. (VAT Group AG, 2017) Dabei übernimmt der Controller die Steuerfunktion dieser Vakuumventile (siehe Abb. 1). Dieser ist modular aufgebaut und besteht grob gesagt aus drei Komponenten. Die zentralen Elemente sind auf dem Masterboard bestückt, welches ein Muss aber angepasst an die jeweilige Ventilhardware ist. Das Herzstück des Controllers ist der Mikrocontroller, für den VAT eine eigene Firmware entwickelt. Weiter ist oder sind Motorbausteine nötig, welche eine weitere Firmware von externen Lieferanten benötigt.

Die zweite Komponente ist das Interface Board. Es wird nach Kundenwunsch angefertigt. Arbeitet der Kunde mit einem Feldbus System, so wird eine Interface Software nötig. Die dritte Komponente ist die Option Unit, die Zusatzfunktionen beinhaltet.



Abbildung : Basiskonzept Ventil Controller (Marugg, 2010)

## Problemstellung

Ich wurde von der Firma VAT eingestellt und beauftragt die Qualität des Controllers, mit Hilfe einer Testumgebung zu überprüfen. Mit dem Programm TTIC2 entwickelte ich eine Testoberfläche, welche einzelne Tests in einer Kollektion zusammenfasst und nacheinander ausführt.

Das Problem ist, dass aktuell nach der Ausführung der Testkollektion das entstandene Report File manuell nach fehlerhaften Testdurchläufen durchsucht werden muss. Das Report File enthält alle Testschritte und erreicht eine sehr grosse Datenmenge. Aus diesem Grund wird meistens nur der letzte Report vor einer Ventil Firmware Freigabe im SVN abgelegt. Dies erschwert die Auswertung von den Tests enorm. Zusätzlich werden auch Zertifizierungsfiles abgelegt, was nochmals erschwerend hinzukommt.

## Ist-Zustand

Das Tool TTIC2, welches die Grundlage dieser Arbeit bietet, wurde eingeführt um Ventil Firmwaren zu qualifizieren. Die Ventiltests sind in CVI geschrieben, da die Hardware über die National Instruments Komponenten sehr einfach anzusprechen sind.

### Stärken

* Alle Funktionalitäten sind auf der Oberfläche ersichtlich (keine Verschachtelungen)
* Die verfügbaren Tests können nach Hardware Eigenschaften gefiltert werden
* Wird ein einzelner Test angewählt, so wird eine Beschreibung des Tests sowie die Hardware Anforderungen angezeigt
* Es können Testkollektionen abgespeichert werden
* Fortlaufender Report wird auf der Oberfläche angezeigt und im Hintergrund in einem Textfile hinterlegt
* Automatische Generierung von Zertifizierungsfiles

### Schwächen

* **Das Programm wird auf mehreren Rechnern ausgeführt**
  + Erschwerte Auswertung der Tests
  + Die abgespeicherten Testkollektionen sind nur auf dem jeweiligen Rechner sichtbar
* **Keinen Verlauf der Testergebnisse der verschiedenen Ventil Firmwaren ersichtlich**
  + Letzter Report File wird im SVN abgelegt
  + Keine schnelle Suche, ob der Testfehler schon einmal aufgetreten ist
  + Fehlermeldung nur im Reportfile ersichtlich
* Keine Sicherstellung des Grundzustandes
* Auslesen der Testergebnisse geschieht im Report File

# Zielsetzung



Abbildung Konzept Masterarbeit

Die Testergebnisse sollen in der SoftwareVersionsDatabase (SQL, Structured Query Language) hinterlegt werden. Diese Datenbank besteht bereits und hat die einzelnen Firmware Inhalte abgespeichert.

Nun soll sie mit folgenden Elementen erweitert werden:

* Name der Testkollektion
* Datum der Ausführung
* Anzahl fehlerhafte Tests
* Resultat der einzelnen Tests
* Bei einem fehlerhaften Test eine kurze Fehlermeldung sowie ein Diagnostikfile

So soll mit folgenden Informationen der Grundzustand des Testablaufs festgelegt werden:

* Ventil Firmware
* Motion Controller Firmware
* Interface Firmware
* Antriebsfile
* Konfigurationsfile

Es soll das Programm ETIC2 (E für Evaluation) erstellt werden, welches eine Übersicht über die Ergebnisse der einzelnen Testkollektionen gibt. Das ETIC2 listet die einzelnen Testkollektionen unter einem definierten Grundzustand auf. Das ETIC2 soll zudem noch eine Report Funktion beinhalten, die zum definiertem Grundzustand alle Ergebnisse rapportiert.

## Quantitative Ziele

* Die SoftwareVersionDatabase muss gleichzeitig Schreibanfragen von vier Benutzern bearbeiten können.
* Jede einzelne Testkollektion muss im ETIC2 zu einem Grundzustand zugeordnet worden sein.
* Der Grundzustand kann nur mit bereits vorhandenen Einträgen in der SoftwareVersionsDatabase definiert werden.

## Qualitative Ziele

* Unter Angabe des Namens des Grundzustandes muss auf Knopfdruck im ETIC2 eine Auswertung aller ausgeführten Testkollektionen mit den dazugehörigen Resultaten aufgelistet werden.
* Das ETIC2 soll eine ausgeprägte Suchfunktion enthalten, welche eine schnelle Suche nach Fehlermeldungen erlaubt.
* Mit dem ETIC2 soll das Resultat der ausgeführten Testkollektion schnell und einfach ersichtlich sein.

# Aufgabenabgrenzung

* Keine Anpassungen der einzelnen Tests
* Die Weiterentwicklung der TTIC2 Applikation ist nicht Teil der Masterarbeit.
  + Ausgenommen ist die Anbindung an die SoftwareVersionDatabase
* Die SoftwareVersionDatabase wird erweitert aber die bestehenden Attribute und Inhalte werden nicht angefasst.
* ETIC2 wird für den internen VAT Verwendungszweck entwickelt und nicht für den kommerziellen Gebrauch konzipiert.

# Vorgehenssystematik

Die verschiedenen Dokumente sowie die Codes werden auf Github abgelegt. Der Grund liegt in dem verteilten Zugriff der Dokumente von verschiedenen Geräten sowie die Versionierung des Codes.

|  |  |  |
| --- | --- | --- |
| **Projektplan** | **Was?** | **Wie?** |
| SoftwareVersionsDatabase Modellierung | Datenbanktabellen und ihre Attribute definieren | Sql Skript erstellen und in SQL Management Studio ausführen |
| TTIC2 Zugriff auf SoftwareVersionsDatabase | Speicherung der Testresultate + Auswahl Grundzustand | Anbindung der SoftwareVersionsDatabase mit Entity Framework |
| Erstellung ETIC2 |  |  |
| * Design View Model | Auslegung der ETIC2 Oberfläche | Präsentationlogik in WPF erstellen |
| * Codierung nach MVVM | Programmierung der ETIC2 Logik | Mit dem Entwurfskonzept MVVM und DevExpress Framework in C# |
| * Anbindung SoftwareVersionsDatabase | Auslesung der Testresultate | Auslesung der SoftwareVersionsDatabase mit Entity Framework |
| * Ausgabe Bericht | Erstellung Testreport in PDF Format | Noch nicht klar |
| * Unit Test | Validierung des ETIC2 | Mit Hilfe des Unit Test Framework |
| Schriftliche Arbeit | Beschreibung der Arbeitsschritte | Wird in Word geschrieben |
| Fertigstellen der schriftlichen Arbeit | Dokumentation der Resultate | Wird in Word geschrieben |

Tabelle Vorgehenssystematik

## Wirtschaftliche Aspekte

In zwei Bereichen kann durch die Erstellung des ETIC2 kontinuierlich Zeit eingespart werden. Erstens ist das Auslesen der einzelnen Testresultate nach Ablauf einer Testkollektion auf einen Blick möglich. Zudem ist die Erstellung eines Testreports auf einen Knopfdruck beschränkt.

Zweitens kann über die Historisierung der Testresultate ein auftretender Fehler besser eingegrenzt und analysiert werden.

## Zeitplan

|  |  |  |  |
| --- | --- | --- | --- |
| **Projektplan** | **Start** | **Ende** | **Aufwand [h]** |
| Einreichen der Disposition | 28.02.2017 | 28.02.2017 |  |
| Start Masterarbeit | 01.05.2017 | 01.05.2017 |  |
| SoftwareVersionsDatabase Modellierung | 01.05.2017 | 05.05.2017 | 15 |
| TTIC2 Zugriff auf SoftwareVersionsDatabase | 08.05.2017 | 21.05.2017 | 30 |
| Erstellung ETIC2 | 22.05.2017 | 13.08.2017 | 155 |
| * Design View Model | 22.05.2017 | 25.05.2017 | 10 |
| * Codierung nach MVVM | 26.05.2017 | 09.07.2017 | 70 |
| * Anbindung SoftwareVersionsDatabase | 26.05.2017 | 09.07.2017 | 5 |
| * Ausgabe Bericht | 10.07.2017 | 16.07.2017 | 20 |
| * Unit Test | 17.07.2017 | 30.07.2017 | 30 |
| * Reserve | 31.07.2017 | 13.08.2017 | 20 |
| Schriftliche Arbeit | 28.02.2017 | 07.08.2017 | 80 |
| Fertigstellen der schriftlichen Arbeit | 14.08.2017 | 31.08.2017 | 40 |
| Abgabe der Masterarbeit | 31.08.2017 |  |  |
|  |  |  |  |
| Total Projektarbeit | 28.02.2017 | 31.08.2017 | 320 |

Tabelle Zeitplan

# (erwartete) Resultate

Zusammenfassend wird als Output erwartet, dass die bestehende SoftwareVersionsDatabase so erweitert wird, dass erstens der Grundzustand jeder Testkollektion definiert werden kann und zweitens auch die Testresultate unter dem Grundzustand aufgelistet werden. Dazu wird das ETIC2 erstellt. Dieses Programm ermöglicht eine schnelle Übersicht über die einzelnen ausgeführten Testkollektionen eines Grundzustandes. Zudem beinhaltet es eine einfache Suche nach Fehlermeldungen. Auf Knopfdruck lässt sich ein Testreport im PDF Format erstellen.

# Glossar

|  |  |
| --- | --- |
| TTIC2 | Test Tool Integrierter Controller 2: Testoberfläche für alle integrierten Ventilcontroller der 2. Generation |
| ETIC2 | Evaluation Tool Integrierter Controller 2. Auswertungsoberfläche für die Testkollektionen |
| IC | Integrierter Controller: Der Controller befindet sich direkt beim Vakuumventil. |
| CVI | Ist eine ereignisorientierte Programmiersprache, welche auf C basiert und von National Instruments entwickelt wurde. . |
| Diagnostik File | Enthält alle Ventilparameter mit ihren aktuellen Werten. Zur genaueren Auswertung eines Fehlers. |
| SVN | Apache Subversion, freie Software zur Versionsverwaltung. |
| Konfigurations-  file | Enthält alle Abweichungen der Software gegenüber den Standard Ventil Firmware Einstellungen, welche in der Firmware hinterlegt sind. |
| Antriebsfile | Enthält alle Ventilhardware spezifische Abweichungen gegenüber den Standard Einstellungen, welche in der Firmware hinterlegt sind. |
| Grundzustand | Der Grundzustand setzt sich aus den Angaben der Ventil Firmware, der Motion Controller Firmware sowie optional der Interface Firmware, des Antriebsfiles sowie Konfigurationsfiles zusammen. Jeder Grundzustand erhält einen eindeutigen Namen. |
| WPF | Windows Presentation Foundation |
| MVVM | Mode View ViewModel |
| DevExpress | Käuflich erworbene Bibliothek für die Verwendung von WPF Elementen |

# Abbildungsverzeichnis und Tabellenverzeichnis

[Abbildung 1: Basiskonzept Ventil Controller (Marugg, 2010) 3](#_Toc475890563)

[Abbildung 2 Konzept Masterarbeit 4](#_Toc475890564)

[Tabelle 1 Vorgehenssystematik 5](#_Toc475890568)

[Tabelle 2 Zeitplan 6](#_Toc475890569)

# Literaturverzeichnis

[1] Marugg, L. (2010). PG\_Info\_Hardware. VAT Interne Präsentation. Haag

[2] VAT Group AG (2017). Geschäftssegmente. Abgerufen am 26.02.2017 unter http://www.vatvalve.com/de/business/industry