Projektarbeit

Abschlussprüfung Winter 2024/2025

Fachinformatiker für Anwendungsentwicklung

Prüfling:

Basel Hlal

Mozartstraße 6

77694 Kehl

Praktikumsbetrieb:

THERMOTEX NAGEL GmbH

Schutterwaldstraße 14

77746 Schutterwald

Entwicklung einer Benutzeroberfläche, die Mitarbeitern in der Werkstatt hilft, den korrekten Anschluss von Komponenten sowie die Ausrichtung und Kalibrierung von Antennen zu überprüfen.

Inhalt

1 Abkürzungsverzeichnis 3

2 Einleitung 4

2.1 Projektumfeld 4

2.2 Projektziel 4

2.3 Projektbegründung 5

2.4 Projektschnittstellen 6

2.5 Ansprechpartner 7

2.6 Projektabgrenzung 7

3 Projektplanung 7

3.1 Projektphasen 7

3.2 Ressourcenplanung 8

3.3 Entwicklungsprozess 9

4 Analysephase 10

4.1 Ist-Analyse 10

4.2 Soll-Analyse 10

4.2.1 Zielsetzung 11

4.2.2 Funktionale Anforderungen 11

4.2.3 Nicht funktionale Anforderungen 12

4.3 Wirtschaftlichkeitsanalyse 13

4.4 Projektkosten 13

4.5 Amortisationsdauer 13

4.6 Anwendungsfall: 14

4.7 Qualitätsanforderungen 15

5 Entwurfsphase 15

5.1 Zielplattform 15

5.2 Architekturdesign 16

5.3 Datenmodell 16

5.4 Benutzeroberfläche 16

6 Implementierungsphase 17

6.1 Implementierung der Benutzeroberfläche 17

6.2 Implementierung der Geschäftslogik 19

7 Testphase 23

8 Abnahmephase 24

8.1 Übergabe 24

8.2 Soll-Ist-Vergleich 24

8.3 Fazit 24

9 Anhang 25

9.1 Übergabeprotokoll 25

9.2 Lastenheft 27

9.3 Pflichtenheft 30

1 Abkürzungsverzeichnis

TTX	THERMOTEX NAGLE GmbH
UI	Benutzeroberfläche (User Interface)
WuT	Web-IO Digital 4.0 (Wiesemann & Theis GmbH)
UHF-R	RFID-Reader (865.7-867.5) MHz / SttID/SIL-9400-MUX4
K-L	RFID-Reader 13.56MHz / TS_HR38 GesmbHs
Antenne	UHF-System Long Range Universal Antenna
UHF-Chips	(865.7-867.5) MHz chips
KL-Chips	13.56MHz Chips

2 Einleitung

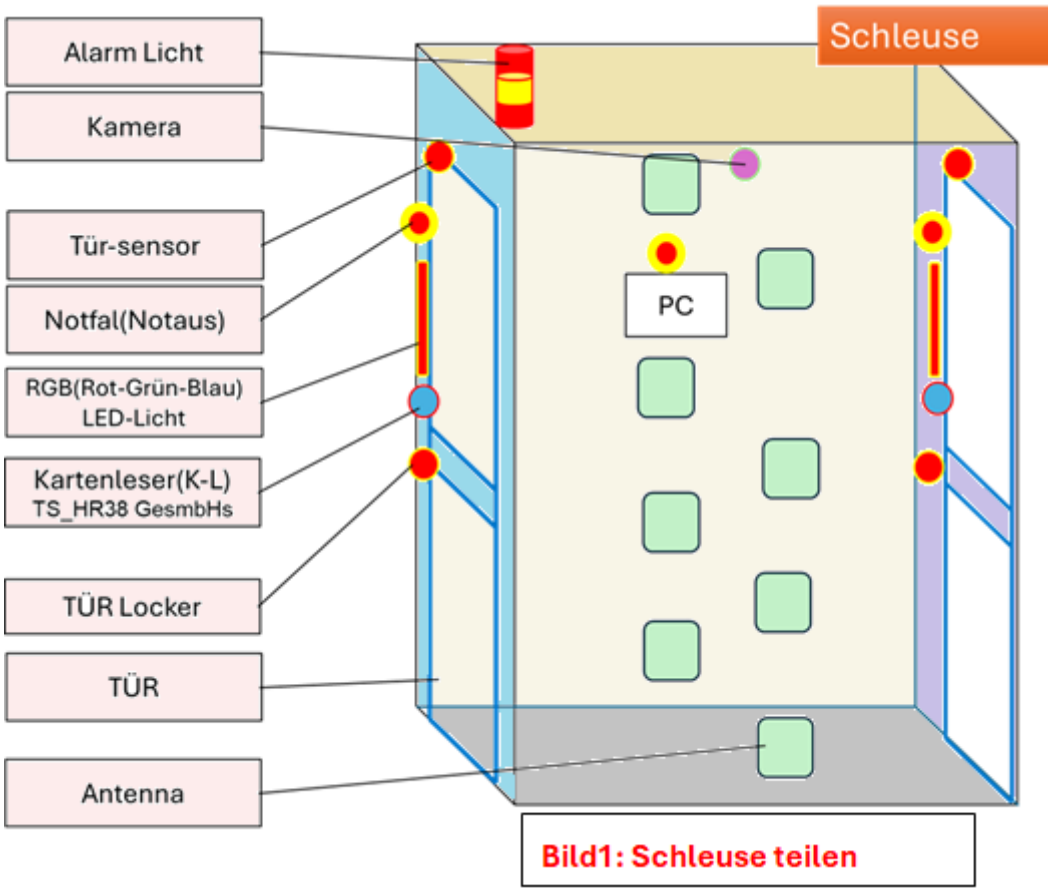
In dieser Dokumentation werden die Vorgehensweise und Umsetzung meines Abschlussprojektes dargestellt. Es handelt sich dabei um ein internes Projekt, welches in meiner Praktikumsfirma THERMOTEX NAGEL GmbH (TTX) umgesetzt wurde.
Die Firma TTX ist auf das Patchen und Kennzeichnen von Wäsche, sowie die Automatisierung und Digitalisierung in Wäschereien spezialisiert.
Hierfür wird die RFID-Technologie eingesetzt, um Wäsche zu identifizieren und eine Echtzeitüberwachung der Wäsche zu ermöglichen.
Diese Technologien werden vor Ort in den Wäschereien installiert.
Die THERMOTEX NAGEL GmbH hat 3 Werke in Schutterwald.

2.1 Projektumfeld

Das Projekt wird in der Abteilung Softwareentwicklung bei TTX entwickelt und in der Programmiersprache Microsoft C# verfasst.

2.2 Projektziel

Eine Schleuse ist ein kleiner Raum(Bild1), der als Durchgang zwischen der waschhalle und dem Außenbereich dient. Diese Schleuse besitzt Sensoren, die automatisch die Türen öffnen und schließen. Zudem hat die Schleuse zwei Arten von RFID-Readern. Einen RFID-Kartenleser, der auf der Frequenz 13.56MHz arbeitet und der Zutrittskontrolle bzw. Identifikation der Mitarbeiter dient. Der andere RFID-Reader, arbeitet auf der Frequenz (865.7-867.5) MHz und dient der Identifikation der Wäsche.
Im Projekt geht es darum, ein TestTool zu entwickeln, das es ermöglicht, noch währen des Aufbaus die Funktionalität der Sensoren in der Personenschleuse
zu testen.



Das TestTool kommuniziert mit WuT, UHF-R und K-L über die Bibliothek PersonSchleuse.dll.

Die PersonenSchleuse.dll enthält zahlreiche Funktionen:

- Herstellen und Trennen der Verbindung zu den Geräten .
- Senden von Befehlen an die Geräte WuT und UHF-R.
- Lesen diese dynamische Bibliothek (DLL) Nachrichten von den Geräten und Weitersenden dieser Nachrichtenmithelfe von Events.
- Senden der Fehlermeldungen, falls ein Fehler auftritt.

2.3 Projektbegründung

Während des Zusammenbaus der Maschine (Schleuse) nutzen die Arbeiter standardisierte Messgeräte, um die Verbindungen der Bauteile zu überprüfen. Zum Schluss wird die Produktivsoftware installiert, um den abschließenden Test durchzuführen.

Hier treten einige Probleme auf:

1. Da der Maschinentest erst nach dem Zusammenbauen gemacht wird, müssen bei einem Fehler einige Teile abgebaut und danach wieder zusammengebaut werden.
2. Weder die grundlegende Produktivsoftware noch die standardisierten Prüfgeräte unterstützen die Ausrichtung der mit dem UHF-R verbundenen Antennen so, dass sie alle UHF-Chips innerhalb der Schleuse lesen können, ohne Chips außerhalb der Schleuse zu erfassen.
3. Die grundlegende Betriebssoftware ist für den Endnutzer ausgelegt, während das Test-Tool speziell für den Techniker entwickelt wurde, der die Schleuse zusammenbaut (Bild2)

Wie ersichtlich, müsste nach dieser Fehlermeldung die Schleuse teilweise demontiert werden. Dies kann durch die Entwicklung eines neuen TestTool,

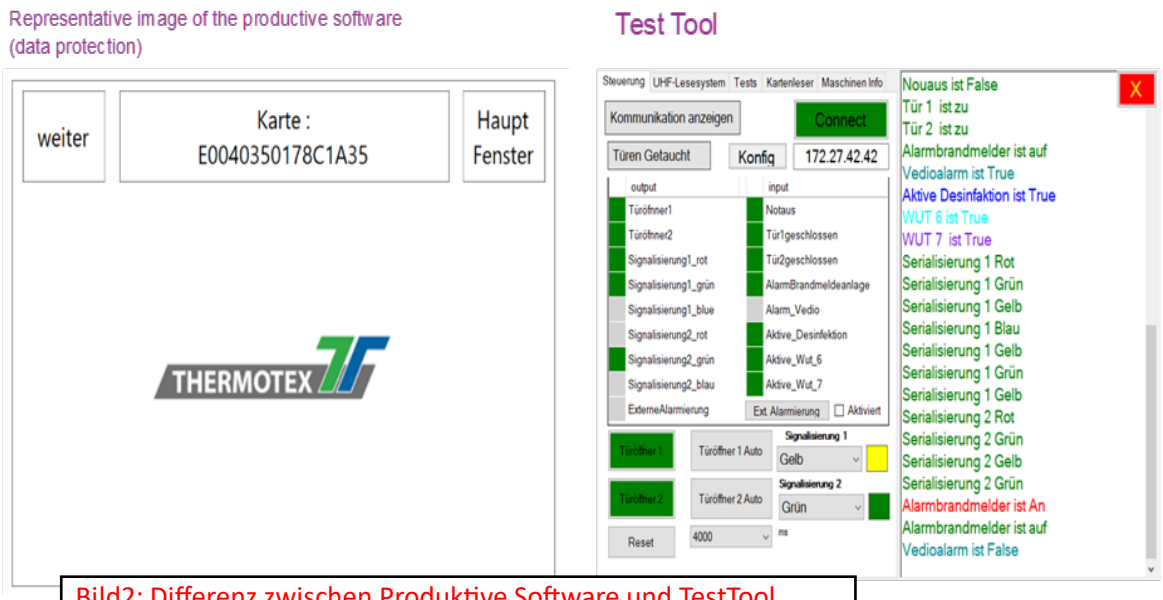


Bild2: Differenz zwischen Produktive Software und TestTool

welches einen Test bereits während des Einbaus ermöglicht, vermieden werden.

2.4 Projektschnittstellen

Die Verbindung zu den Geräten erfolgt wie folgt:

- WuT: Verbindung über TCP/IP
- UHF-R: Verbindung über TCP/IP und COM.
- K-L: Verbindung über COM

Das Programm wird ausschließlich während des Zusammenbaus der Schleuse oder im Falle von Wartungsarbeiten verwendet.

2.5 Ansprechpartner

Im Projekt dienten folgende Personen als Ansprechpartner

Teamleiter: Reisigl, Eduard-Florian

Teamkollegin: Dinger, Katja

Elektrotechniker: Teichfischer, Jörg

2.6 Projektabgrenzung

Das Programm darf nicht auf die, in manchen Modellen verbaute Kamera zugreifen, sondern lediglich den Status des Kamerasensors (ON/OFF) abrufen, wenn dieser konfiguriert ist. Dieser Sensor ist über Input 5 mit der Wut verbunden.

3 Projektplanung

3.1 Projektphasen

Ein grober Zeitplan für die Arbeitsaufgaben:

1	Analyse	4std
	Ermittlung des Ist-Zustand	2
	Studieren des Lastenheftes	2
2	Entwurfsphase	9std
	Erstellung des Pflichtenheftes	3
	Erstellung der Zeitplanung	3
	Personal, Sachmittel und Kostenplanung	1
	Planung der Anwendung/Hardware	2
3	• Entwicklung und Implementieren	53std
	Implementierung Web-IO	8
	Implementierung RFID_Reader	8
	Implementierung Leak-TestTool	7
	Implementierung K-L	8

	Implementierung Datenbank	8
	UI-Verbessern	8
	Primäre Tests	6
4	Abschlusstests	4std
	Finale Test	2
	Übergabe	1
	Soll-Ist-Vergleich	1
5	Projektdokumentation	10std
	Projekt Dokumentation	8
	Kundendokumentation	2
Gesamt		80std

Tabelle1: Projekt planen

3.2 Ressourcenplanung

Das Projekt wurde am firmeninternen PC-Arbeitsplatz durchgeführt.

- WuT

  Web-IO Digital 4.0

  Von W&T (www.Wut.de)

- UHF-R

  Oder RFID-reader from Scemtec

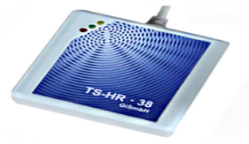
  Transport Technology GmbH (Stt)

- RF-ID Antenna

  UHF-System Long Range Universal

- (865.7-867.5) MHz chips

  Es wird von UHF-R Gelesen

• Kartenleser TS-HR – 38 GiSmbH	
• 13.56MHz Chips Es wird von K-L Gelesen	
Tabelle 2: Schleuse Haupt geräten	

Folgende Software kam im Projekt zum Einsatz:

- Betriebssystem: Windows 10
- Programmiersprache: C# mit Visual Studio 2022 Professional

Bei Programmierung wurden diese Bibliotheken verwendet:

- PersonenSchleuse.dll, die von TTX entwickelt wurde

3.3 Entwicklungsprozess

Die meisten Maschinen, die von TTX Zusammen gebaut werden, benötigen die TestTool Software.

Zur Umsetzung des Projekts wurde dieses in neun Aufgaben unterteilt, wobei jede Aufgabe einen Arbeitstag in Anspruch nahm.

Jeder Arbeitstag begann mit einer maximal 15-minütigen Besprechung, in der die Ergebnisse des Vortages besprochen und die Aufgaben für den aktuellen Tag geplant wurden.

Da der Entwickler allein arbeitete, war es sinnvoll, nach Abschluss einer Phase direkt zur nächsten Phase überzugehen.

Somit kann die Arbeitsweise als an das Wasserfallmodell angelehnt beschrieben werden.

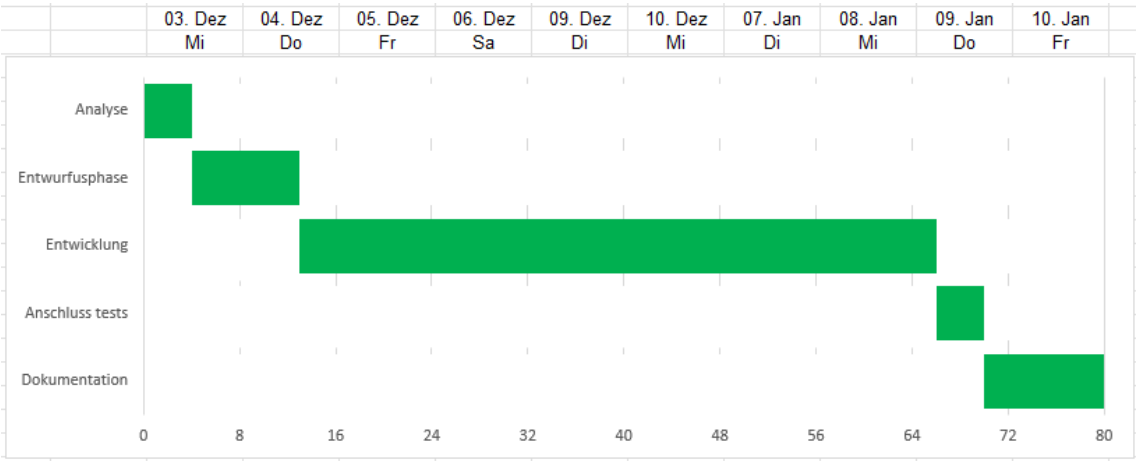


Bild 3 : Arbeitsablauf WasserfallModell

4 Analysephase

4.1 Ist-Analyse

Derzeit werden die Hardware-Komponenten in der Werkstatt manuell verdrahtet und mithilfe von Messgeräten auf ihre korrekte Verkabelung überprüft. Ein realistischer Test der Hardware ist jedoch erst nach dem vollständigen Zusammenbau und unter Einsatz der Produktivsoftware möglich (Bild2). Dieses Vorgehen mag während der Entwicklungsphase als Übergangslösung akzeptabel sein, ist jedoch für die Serienproduktion ungeeignet. Die Fehlersuche und -behebung gestaltet sich auf diese Weise äußerst komplex und zeitintensiv.

4.2 Soll-Analyse

Um die korrekte Verkabelung sowie die Funktionsfähigkeit der eingebauten Hardware-Komponenten (UHF-R, Chipkartenleser, Multi-IO-Port) sicherzustellen, wird ein benutzerfreundliches Testwerkzeug benötigt. Der erforderlichen DLL zur Kommunikation mit der Hardware sind bereits vorhanden. Es soll eine einfache Anwendung entwickelt werden, die es Montage- und Servicetechnikern ermöglicht, die grundlegende Funktionalität der Hardware-Komponenten schnell und unkompliziert zu testen.

4.2.1 Zielsetzung

Das Ziel des Projekts ist die Entwicklung einer **Software** zur Überprüfung der Funktionalität von **Hardwarekomponenten**. Die Software soll folgende Anforderungen erfüllen:

1. **Funktionale Anforderungen**:
   - Die Software muss in der Lage sein, die Funktionalität der verschiedenen Hardwarekomponenten präzise zu prüfen.
   - Die Testergebnisse müssen korrekt und zuverlässig sein.
2. **Benutzerfreundlichkeit**:
   - Die Software soll für Werkstattmitarbeiter leicht verständlich und bedienbar sein.
3. **Effizienzsteigerung**:
   - Die Nutzung der Software soll zu einer **Zeitersparnis** führen.
   - Verbesserung der **Qualität** der Prüfprozesse.
4. **Technische Anforderungen**:
   - Integration von **DLLs**, die in C# für die einzelnen Hardwarekomponenten erstellt wurden.
5. **Projektplanung**:
   - Die Software muss bis zum **17.12.2024** fertiggestellt werden.
   - Das **Budget** für die Entwicklung beträgt **1500 €**.
6. **Weitere Aspekte**:
   - Das Projekt läuft nicht auf eine Ausschreibung hinaus.

4.2.2 Funktionale Anforderungen

Erforderliche Funktionen:

1. **Testen der Hardwarekomponenten**:
   - Die Software soll in der Lage sein, alle Funktionen der Hardwarekomponenten zu prüfen.
2. **Anzeigen der Testergebnisse**:
   - Die Ergebnisse der durchgeführten Tests sollen klar und verständlich angezeigt werden.

Nicht-Erforderliche Funktionen:

1. **Keine Erweiterbarkeit**:
   - Das Produkt muss nicht erweiterbar sein, d.h., es ist nicht vorgesehen, die Software nachträglich um zusätzliche Funktionen zu ergänzen.

4.2.3 Nicht funktionale Anforderungen

1. **Bedienbarkeit:**
   - Die Software muss **einfach und schnell zu bedienen** sein, sodass Werkstattmitarbeiter ohne große Schulung effektiv damit arbeiten können.
2. **Erweiterbarkeit:**
   - Das Produkt muss nicht erweiterbar sein. Zukünftige Funktionserweiterungen sind nicht vorgesehen.
3. **Änderbarkeit:**
   - Änderungen am Produkt sind **nicht möglich**, weder in der Funktion noch in anderen Aspekten.
4. **Standzeiten:**
   - Es gibt keine Anforderungen an spezifische Standzeiten für das Produkt.
5. **Wartungsintervalle:**
   - Keine Angaben; möglicherweise ist keine regelmäßige Wartung erforderlich.
6. **Zuverlässigkeit:**
   - **Volle Zuverlässigkeit** wird gefordert. Die Software muss unter allen vorgesehenen Bedingungen korrekt funktionieren.
7. **Toleranzen:**
   - **Keine Toleranzen** werden akzeptiert, d.h., die Testergebnisse müssen präzise und fehlerfrei sein.
8. **Effizienz:**
   - Das Produkt muss eine **Steigerung der Effizienz** beim Testen der Funktionalität der Hardwarekomponenten gewährleisten, indem es schneller und zuverlässiger arbeitet als bisherige Methoden.

4.3 Wirtschaftlichkeitsanalyse

Mit der **TestTool Software** kann der Mitarbeiter in der Werkstatt (Monteur) die Verbindungen schnell und effizient überprüfen.

Bei auftretenden Fehlern zeigt das Programm gezielte Meldungen an, die bei der Fehleranalyse und Problemlösung unterstützen.

Ein weiterer Vorteil ist, dass der Testprozess der Maschine bereits während des Zusammenbaus beginnt und nicht erst nach dessen Fertigstellung. Dies trägt zur Reduktion der Bauzeit bei.

4.4 Projektkosten

Da keine zusätzlichen Anschaffungen von Software oder Hardware erforderlich waren und alle verwendeten Bibliotheken intern entwickelt wurden, beschränkten sich die Projektkosten auf die Gehälter des Entwicklers und des Teamleiters, der das Projekt betreute.

Mitarbeiter	Stundenzahl	Stundensatz	Gesamtkosten
Teamleiter	(10 * 15min) = 2.5 Std	40 €/Std	100 €
Entwickler	80 Std	18 €/Std	1440 €
Summe			1450 €

Tabelle1: Kostenplanung

Die Gesamtkosten des Projektes betrugen 1450€. Die geplanten Projektkosten liegen damit 50 € unter dem vom Auftraggeber vorgegebenen Budget von 1500 €.

4.5 Amortisationsdauer

Die Kosten pro Stunde für den Bau einer Schleuse setzen sich aus den Kosten für den Techniker mit 33 €, für einen Hilfsarbeiter mit 19€ und den Servicekosten (Strom, Wasser, etc.) mit 18 € zusammen und betragen somit 70€.

Durch die Nutzung der TestTool Software wird bei der Montage einer PersonenSchleuse durchschnittlich 1 Stunde Arbeitszeit für das gesamte Team eingespart. Das bedeutet eine Ersparnis von **70 Euro** pro Stück.

$$\boldsymbol{Anzahl\ der\ Schleusen} = \frac{1450}{70} = 20.71 \cong 21\ \boldsymbol{Stück}$$

Nach dem Bau von 21 Schleusen hat sich die Eigenentwicklung des neuen TestTool amortisiert.

4.6 Anwendungsfall:

(Z.B.) Überprüfung der WuT

Ziel: Überprüfung der korrekten Verbindungen zwischen den Geräten Wut über das TestTool (Bild 4).

Akteure: Der Arbeiter.

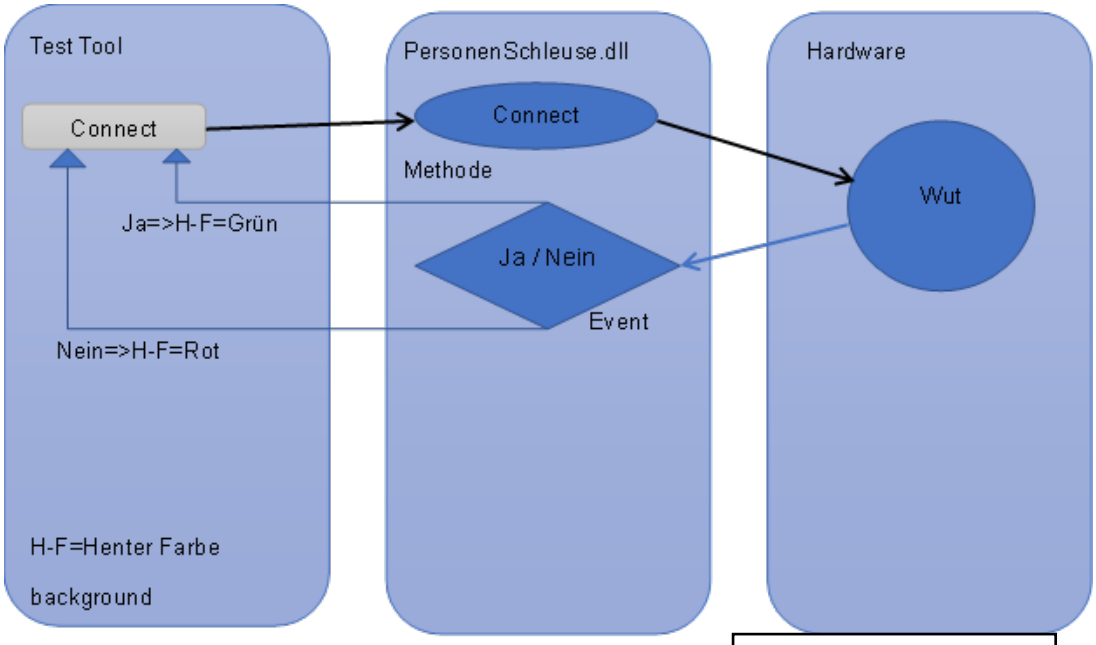


Bild 4: connect(uml)

Hauptablauf: Steht in diesem Komponenten-Diagramm mit Datenfluss

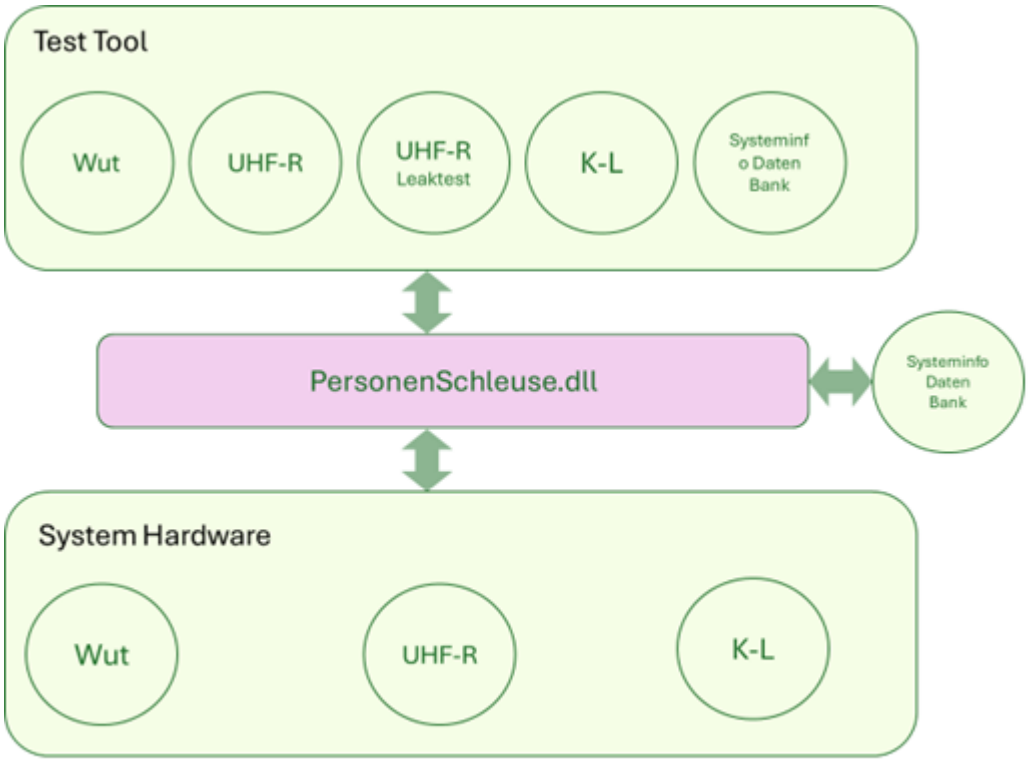


Bild5: Systemarchitektur-Diagramm

4.7 Qualitätsanforderungen

Die in diesem Programm geforderten Qualitätsanforderungen sind:

- Die Benutzeroberfläche muss **einfach** und **umfassend** gestaltet sein.
- Eine **automatische Aktualisierung** der Hardware-Informationen muss gewährleistet sein.
- Es sollte die Möglichkeit geben, die **Hardware** zu senden.

5 Entwurfsphase

5.1 Zielplattform

- Betriebssystem: Windows10
- Programmiersprache und Framework: C# mit .NET 4.8
- (Minimum) Prozessor Core 2 GHz, Ram 4GB, Festplatte,
- Touch-Screen 10.4 inch (210mm x 160mm)

5.2 Architekturdesign

Da die **PersonenSchleuse.dll** in **C# mit .NET 4.8** geschrieben wurde, stehen die folgenden Optionen zur Verfügung: **Visual Basic** und **C#**.

Visual Basic eignet sich für kleine, kurzfristige Projekte, während C# für moderne Projekte besser geeignet ist, da es umfangreiche Unterstützung von Microsoft erhält und gut mit der .NET-Umgebung kompatibel ist.

Daher habe ich mich für C# entschieden.

5.3 Datenmodell

Die **PersonenSchleuse.dll** arbeitet mit einer **SQLite-Datenbank**, aber das **TestTool** greift nicht direkt auf Datenbanken zu. Stattdessen zeigt es eine Tabelle an, die von der **PersonenSchleuse.dll** übermittelt wird.

5.4 Benutzeroberfläche

Die Benutzeroberfläche wird in zwei Bereiche unterteilt

- Enthält eine **LOG-Fenster** (RichTextBox Tool), die alle Ereignisse, die von der **PersonenSchleuse.dll** gesendet werden, in **textlicher und farblicher Form** darstellt.
- Zeigen die **gleichen Informationen** wie im ersten Bereich, jedoch in **grafischer Form**. Zusätzlich enthält dieser Bereich **Steuerungselemente (Buttons)** für die Hardware, wie:
  - WuT verbinden /trennen
  - Tür (1/2) Auf/zu machen
  - LED-Licht An/Auf (Rot, Grün, Blau, Gelb, Rotblinken)
  - Alarm An/Aus machen
  - UHF-Reader verbinden/Trennen (durch LAN oder COM)
  - RFID-Chips Scannen
  - Leaktest Machen
  - K-L verbunden / trennen
  - Datenbank anschauen, zurücksetzen

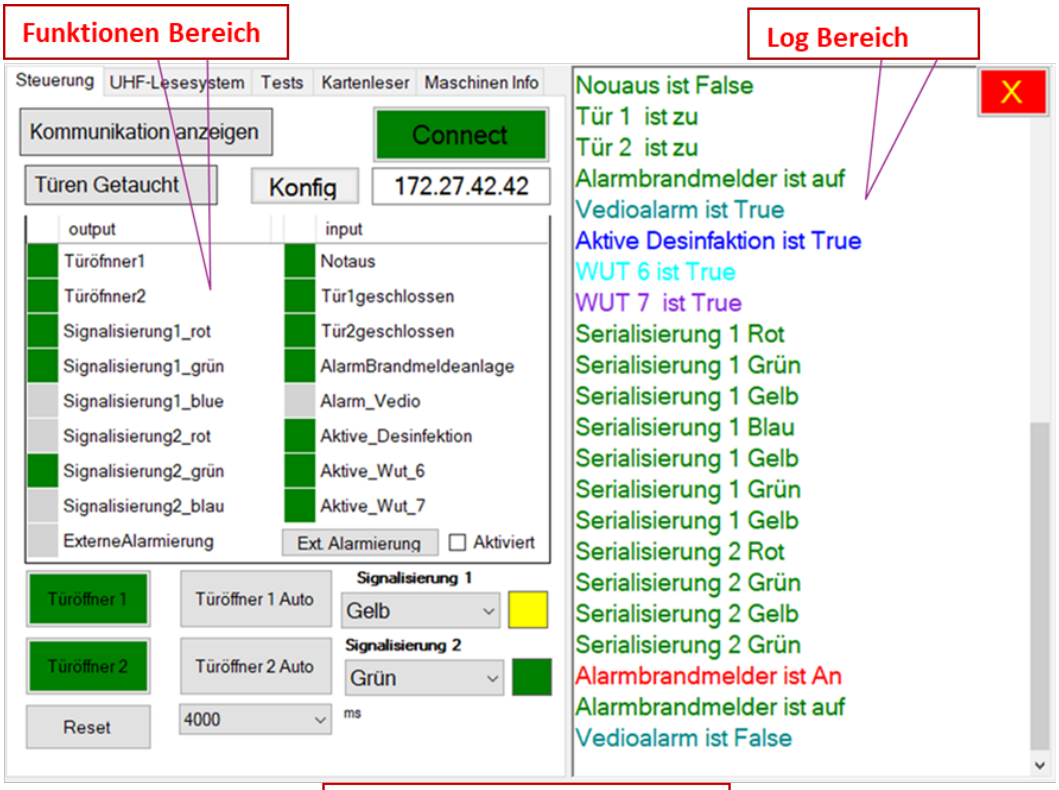


Bild6 : Benutzeroberflache

6 Implementierungsphase

6.1 Implementierung der Benutzeroberfläche

Es gibt ein Log und einen Funktionen Bereich in der Benutzeroberfläche.

Im **Logbereich** kann man alle Ereignisse lesen, die im System stattgefunden haben – von der Startzeit bis zum aktuellen Zeitpunkt. Man kann den **Scrollbalken** verwenden, um frühere oder neuere Ereignisse anzuzeigen. Es ist jedoch **nicht möglich**, das Logfenster zu nutzen, um **Befehle an die Hardware** oder an **PersonenSchleuse.dll** zu senden.

Im Funktionenbereich Bereich gibt es ein **TabControl**, das in fünf Abschnitte unterteilt ist, die in den Bildern dargestellt sind.

Der Funktionenbereich kann eine folgenden sein:

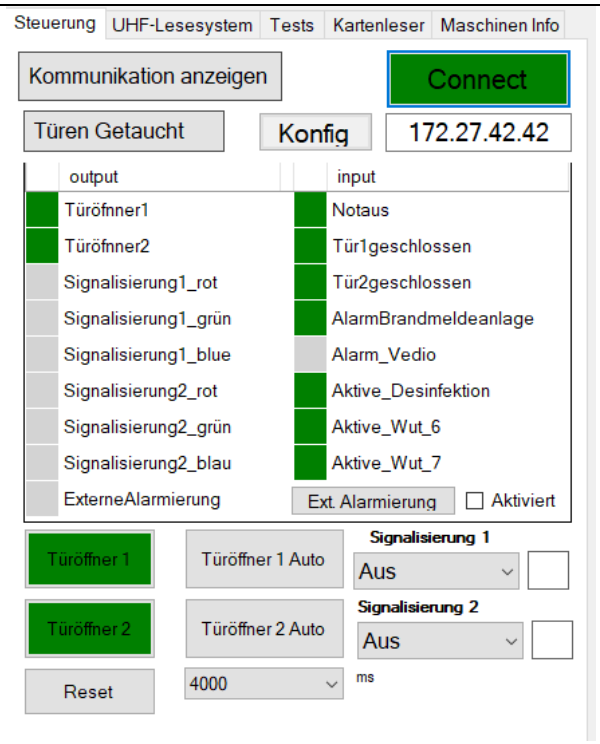


Bild A1: WuT

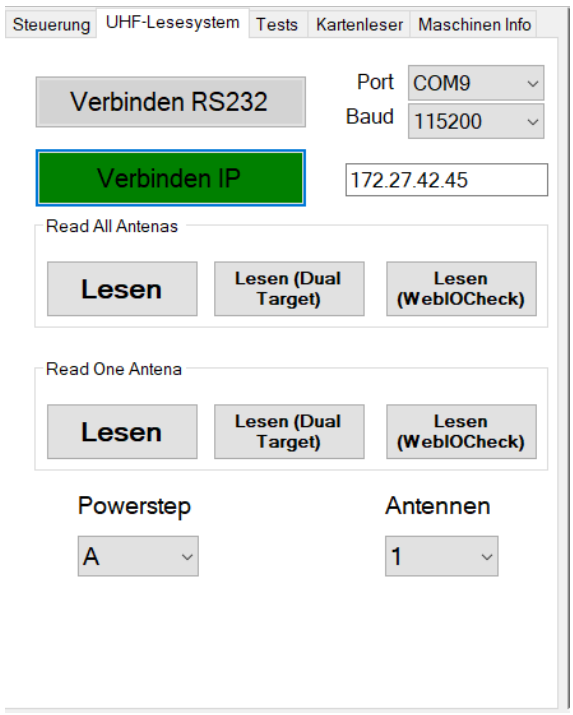


Bild A2: UHF-R

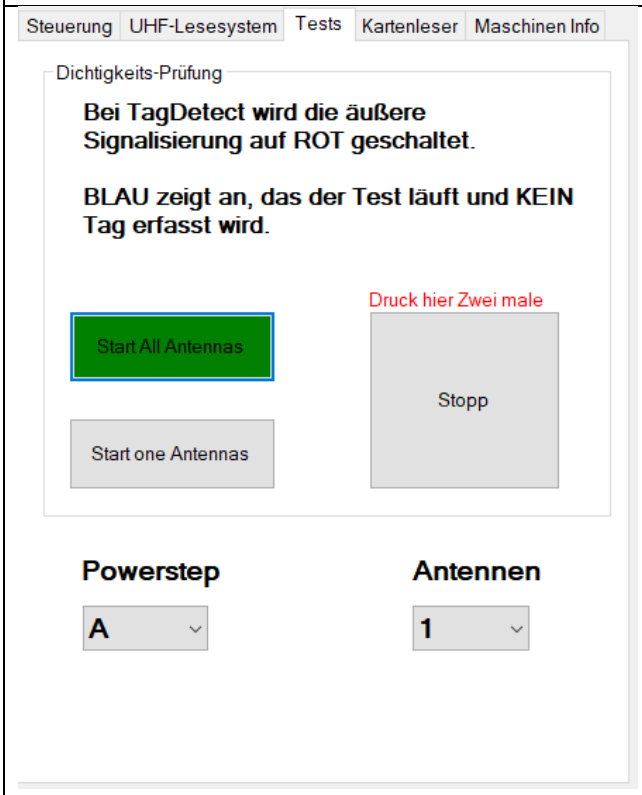


Bild A3: Leaktest

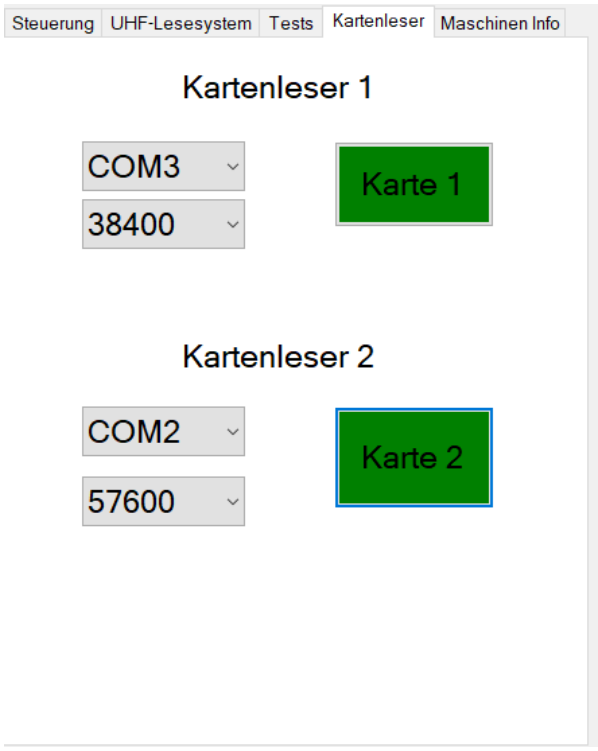


Bild A4: Kartenleser

Bild A5: Maschinen Infos

6.2 Implementierung der Geschäftslogik

In einem vorherigen Projekt habe ich die **PersonenSchleuse.dll** entwickelt. Dabei wurden Methoden für alle erforderlichen Funktionen zur Kommunikation mit der Hardware geschrieben. Zudem gibt es **Events**, die Nachrichten über jede Änderung in der Hardware senden, z. B:

Code Türöffner1

```
1 reference
private void Bt_Türöffner1_Click(object sender, EventArgs e)
{
    wut.Door1_Open(false);
}
```

Bild 7 : Türe 1 zu machen (False)

Events Definition Kode

```
rf_reader.LeakTeast_event += Rf_reader_LeakTeast_event;              //return (bool, string)
rf_reader.read_something_event += Rf_reader_read_something_event;    //return List<string>
rf_reader.Error_messag_Event += Rf_reader_Error_messag_Event;        //return string error me
rf_reader.isconneced_event += Rf_reader_isconneced_event;            //return bool connected=

wut._wut_Is_Connected_event += Wut__wut_Is_Connected_event; //return connected=true,disconnect
wut.Error_event += Wut_Error_event;                         //return string error.message
wut.get_Outputs_Event += Wut_get_Outputs_Event;             //return bool[12] array , on=true,
wut.Message_event += Wut_Message_event;                     //return string like "Connected" o
wut.NoutAus_Event += Wut_NoutAus_Event;                     //return bool On=true , Off=false
wut.Tür1Sensore_event += Wut_Tür1Sensore_event;             //return bool On=true , Off=false
wut.Tür2Sensore_event += Wut_Tür2Sensore_event;             //return bool On=true , Off=false
wut.AlarmBrandmelder_event += Wut_AlarmBrandmelder_event;   //return bool On=true , Off=false
wut.AlarmVedio_event += Wut_AlarmVedio_event;               //return bool On=true , Off=false
wut.Aktive_Desinfektion_event += Wut_Aktive_Desinfektion_event; //return bool On=true , Off=fa
wut.wut6_event += Wut_wut6_event;                           //return bool On=true , Off=false
wut.wut7_event += Wut_wut7_event;                           //return bool On=true , Off=false
wut.new_DataTable_Event += Wut_new_DataTable_Event;         //return datatable of meschinen in

c_reader1.Data_Error_Event += C_reader1_Data_Error_Event;                 //return string er
c_reader2.Data_Error_Event += C_reader2_Data_Error_Event;                 //return string er
c_reader1.connection_state_event += C_reader1_connection_state_event;     //return string er
c_reader2.connection_state_event += C_reader2_connection_state_event;     //return string er
c_reader1.Data_Received_Event += C_reader1_Data_Received_Event;           //return ReaderRes
c_reader2.Data_Received_Event += C_reader2_Data_Received_Event;           //return ReaderRes
```

Bild 8: Events Definieren

Jedes **Event** ruft eine einfache **Methode** auf, die eine Nachricht auf der **Nachrichtentafel** schreibt.

Zudem wird eine pass ende Aktion im **Funktionen Bereich** ausgeführt.

```
1 reference
private void Wut_Tür1Sensore_event(bool obj)
{
    Log("Tür 1  ist " + (obj ? "zu" : "auf"), Green_Red(obj));
}
```

Bild 9: Eventscode Beispiel

Der Methode Log schreibt im Log Fenster „Tür 1 ist zu/auf“ in Grün oder Rot.

Das bedeutet jedoch nicht, dass es keinen wichtigen Code innerhalb des Programms gibt.

```
1 reference
private void Rf_reader_LeakTeast_event(bool obj,string arg)
{
    this?.Invoke((Action)(() =>
    {  try
        {
            if (laststate == obj) return;
            laststate = obj;
            if (obj)
            {                                                   // find some thing
                Log(">>>  TagDect beim Ant "+arg, Color.Red);   // schriftliche message
                GroupBox1.BackColor = Color.Red;                //visual message (backcolor red)
                wut.Serling_1(1); wut.Serling_2(1);             //Hardware message (LED Red)
            }
            else
            {                                                   // find nothing => reset
                GroupBox1.BackColor = Color.Transparent;        //visual message (Backcollor normal)
                wut.Serling_1(4); wut.Serling_2(4);             // Hardware LED Blue
            }
            Application.DoEvents();
        }
        catch { }
    }));
}
```

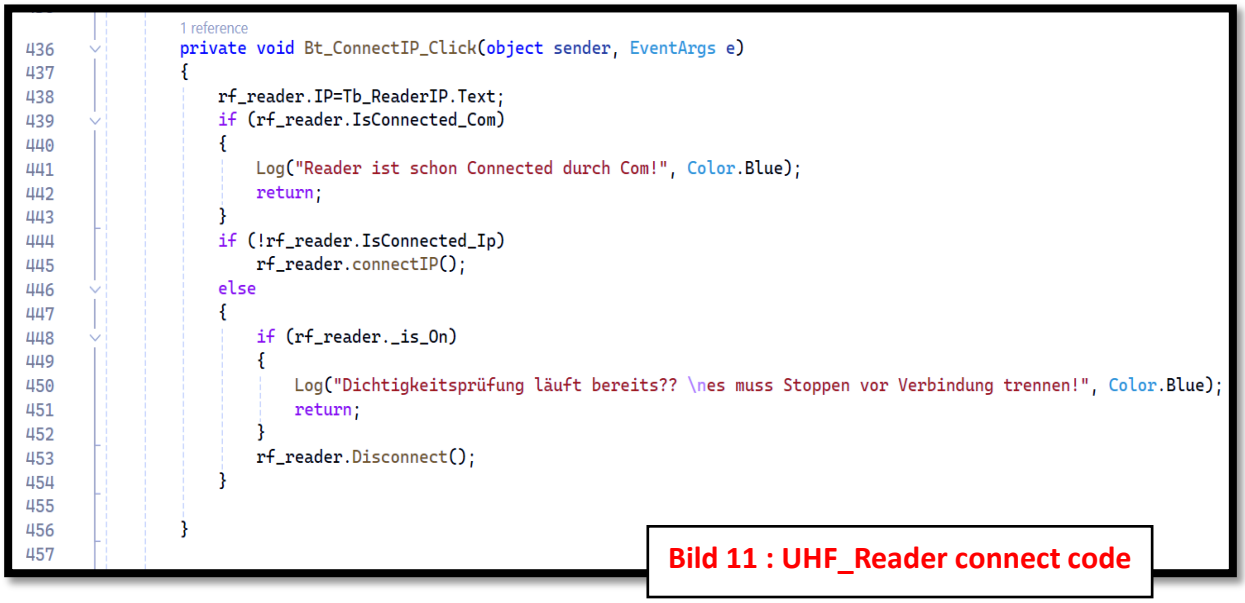
Bild 10: LeakTest Event code

Dieser Code läuft während des Leaktests. Wenn man den Befehl ausführt (durch Drücken der Buttons „Start all Antennas“ oder „Start one Antenna“ im BildA3), führt das UHF-R-System eine Reihe von Ereignissen (Events) aus. Jedes Event enthält zwei Variablen:

- Eine Variable mit einem Boolean-Wert („true“ oder „false“).
- Den Text des Codes vom erkannten Chip.

Funktionsweise des Codes:

- Laststate: Das ist der alte Wert von obj.
- Wenn der alte Wert gleich dem neuen Wert ist, bedeutet das, es gibt keine Änderung, und der Code macht nichts weiter (er kehrt mit return zurück).
- Wenn obj = true (ein Chip wurde erkannt):
- S chreibe den Text „>>> TagDect beim Ant + Chip-Kode“ in rote Farbe ins Logfenster.
- Ändere den Hintergrund der groupBox1 zu Rot.
- Die LED-Anzeige leuchtet Rot.
- Wenn obj = false (kein Chip wurde erkannt):
- Der Hintergrund der groupBox1 wird wieder auf die normale Farbe zurückgesetzt.
- Die LED-Anzeige leuchtet Blau.



```
1 reference
436     private void Bt_ConnectIP_Click(object sender, EventArgs e)
437     {
438         rf_reader.IP=Tb_ReaderIP.Text;
439         if (rf_reader.IsConnected_Com)
440         {
441             Log("Reader ist schon Connected durch Com!", Color.Blue);
442             return;
443         }
444         if (!rf_reader.IsConnected_Ip)
445             rf_reader.connectIP();
446         else
447         {
448             if (rf_reader._is_On)
449             {
450                 Log("Dichtigkeitsprüfung läuft bereits?? \nes muss Stoppen vor Verbindung trennen!", Color.Blue);
451                 return;
452             }
453             rf_reader.Disconnect();
454         }
455
456     }
457
```

Bild 11 : UHF_Reader connect code

1. Der Code wird ausgeführt, wenn der Button **Bt_ConnectIP(**Verbenden IP**)** gedruckt wird(BildA2).
2. Die **IP-Adresse des UHF-R** wird aus der Textbox (Tb_ReaderIP) gelesen.
3. Wenn der UHF-R bereits über COM verbunden ist:
   - Schreibe ins Log-Fenster: **„Reader ist schon Connected durch COM!“** (in blauer Farbe).
   - Der Code wird an dieser Stelle beendet.
4. Wenn der UHF-R nicht verbunden ist:
   - Stelle eine Verbindung über **ConnectIP** her.
5. Falls LeakTest gerade läuft:
   - Schreibe ins Log: **„Dichtigkeitsprüfung läuft bereits…“**.
   - Der Code wird beendet, ohne die Verbindung zu trennen.
6. In allen anderen Fällen:
   - Trenne die Verbindung (Disconnect).

Das Programm hat nicht richtig funktioniert, obwohl der Code korrekt war. Es konnte die Nachrichten, die über Events gesendet wurden, entweder nicht lesen oder hat sie falsch gelesen. weil die Methoden im Hintergrund laufen. Um das Problem zu lösen, musste die Methode „**Invoke**“ benutzt werden, damit die Hintergrundprozesse richtig angezeigt werden.

```
1 reference
private void Wut_Message_event(string obj)
{
    this.Invoke((Action)(() =>Log(obj,Color.Green) ));
}
```

Bild12: Invoke Befehl

- Aufgrund des kleinen Bildschirms (10,4 Zoll) waren die Beschriftungen der Schaltflächen und die Schriftgröße so klein, so dass ein normaler Mensch sie nur schwer lesen konnte.
- Daher wurde die Schriftgröße so weit wie möglich vergrößert, und die Steuerleiste wurde entfernt. Zusätzlich wurden die Schaltflächen und die Texte farblich hervorgehoben, um sie leichter zu unterscheiden:
  - Fehler- oder Schließnachrichten wurden rot eingefärbt.
  - Erfolgsnachrichten wurden grün eingefärbt.
  - Normale Operationen wurden blau markiert.
  - Eine spezielle Farbe für jede Eingabe (Input) in Wut

7 Testphase

Die Tests wurden in zwei Phasen durchgeführt:
während der Programmerstellung und nach der Fertigstellung des Programms.

- (WuT + UHF-R+K-L) wurden an den PC verbunden. Jede Funktion wurde direkt nach ihrer Programmierung getestet, sodass sowohl die Funktion selbst als auch ihre Kompatibilität mit anderen Funktionen überprüft wurden.
- Das Programm wurde an der Schleuse getestet und vom Arbeiter in Anwesenheit des Abteilungsleiters geprüft.
- Es ist erwähnenswert, dass das Testen des Programms gleichzeitig auch einen Test der dynamischen Verknüpfungsbibliothek PersonenSchleuse.dll darstellt.

8 Abnahmephase

8.1 Übergabe

Nach der Prüfung des Programms durch den Techniker (Endbenutzer) und der Anpassung des Programms an die Anforderungen die Benutzer, wurde das Programm dem Abteilungsleiter in Anwesenheit des Teamleiters vorgestellt. Es wurde sichergestellt, dass das Programm alle Funktionen effizient und ohne Probleme ausführt. Daher wurde das Programm akzeptiert und offiziell in Betrieb genommen

8.2 Soll-Ist-Vergleich

Das Projekt wurde innerhalb des Zeitplans umgesetzt und das geplante Budget eingehalten. Alle Leistungsziele wurden so erreicht, dass sie die volle Zufriedenheit des Abteilungsleiter TTX gewährleisten.
In allen Schritten wurden vor Ablauf der vorgegebenen Frist abgeschlossen. Daher kam es zu keinen Abweichungen vom geplanten Betrieb.

8.3 Fazit

Während der Ausbildungszeit im Allgemeinen und durch dieses Projekt im Besonderen habe ich Folgendes gelernt:

1. Klar verständliche Namen zu vergeben, die die Funktion der Variablen im Detail beschreiben.

   Bei spiel:

   ```
   for (int i = 0; i < 9; i++)
   ```

   ```
   for (int IndexOfInput_Wut = 0; IndexOfInput_Wut < Input_Wut.Length; IndexOfInput_Wut++)
   ```

   Dies ist hilfreich für jeden Entwickler, der das Programm weiterentwickeln möchte, und auch für mich selbst, falls ich in Zukunft Änderungen vornehmen möchte.
2. Kommentare im Code zu schreiben, um den Code verständlicher zu machen.
3. Dokumentation (Kommentaren im Code und einer PDF-Datei mit Methoden und Variablen im Code sowie einem Benutzerhandbuch für den Endbenutzer)

4. Die Bedeutung von Koordination und kontinuierlicher Kommunikation mit den Team Mitgliedern zu verstehen und anzuwenden.

9 Anhang

9.1 Übergabeprotokoll

PROJEKTNAME

TestTool

PROJEKTSPONSOR	PROJEKT MANAGER	PROJEKTSTARTDATUM	PROJEKT ENDDATUM
Thermotex Nagel GmbH	Eduard-Florian Reisigl	03.12.2024	10.01.2024

KURZFASSUNG DES PROJEKTS

Ein einfaches Programm, das Zusammenbauarbeitern hilft, die Verbindungen der Komponenten während des Zusammenbaus zu überprüfen

PROJEKTROLLEN UND -VERANTWORTLICHKEITEN

NAME	ROLLE	VERANTWORTLICHKEITEN
Reisigl, Eduard-Florian	1-Teamleiter	Koordinator, Ausbilder
Dinger, Katja	2- Programmierer	Anwendungsentwickler
Teichfischer, Jörg	3- Elektrotechniker	Konstruktion Anlage
Hlal, Basel	4- Programmierer	Anwendungsentwickler

LEISTUNGEN

GEPLANT	TATSÄCHLICH	KOMMENTARE
10 Tage von 03.12.24 bis 10.01.2025	10 Tage von 03.12.24 bis 10.01.2025	Das Projekt wurde im vorgegebenen Zeitrahmen fertiggestellt, jedoch aufgrund von Feiertagen verschob sich der Termin nach hinten

AUSGABEN

PROJEKTPHASE	GEPLANTES BUDGET	TATSÄCHLICHE KOSTEN	KOMMENTARE
Analyse	132€	132€	4*18 Entwickler + 10*40 Teamleiter
Entwurfsphase	162€	162€	19*18
Entwicklung und Implementieren	954€	954€	53*18
Abschlusstests	112€	112€	4*18+1*40
Projektdokumentation	90€	90€	5*18
Gesamt	1450€	1450€	

ZEITPLAN

WICHTIGE MEILENSTEINE	URSPRÜNGLICHE FRIST	TATSÄCHLICHER ABSCHLUSS	KOMMENTARE
Analyse	03.12.24	03.12.24	
Entwurfsphase	03.12.24	04.12.24	
• Entwicklung und Implementieren	04.12.24	09.01.25	

ZEITPLAN

WICHTIGE MEILENSTEINE	URSPRÜNGLICHE FRIST	TATSÄCHLICHER ABSCHLUSS	KOMMENTARE
Abschlusstests	09.01.25	09.01.25	
Projektdokumentation	09.01.25	10.01.25	

9.2 Lastenheft

THERMOTEX	Anfrage / Lastenheft	Projekt-Nr: 12/2024

Projekt-Bezeichnung:	Hardware-TestTool				
Ersteller:	ERE/KDI	Stand:	2	Datum:	03.12.2024
Interessierte Partei:	Thermotex: SW-EW, Werkstatt, Maschinen-EW				

Die Fragen müssen je nach Produkt / Projekt angepasst oder ergänzt werden!

Ausgangssituation

Anforderung / Beschreibung	Parameter
Wie kam es zur Projektidee?	Einfache Möglichkeit für die Mitarbeiter der Werkstatt Hardware Komponenten auf Ihre Funktionalität zu prüfen
Welches Problem ist aufgetreten?	Das Testen von UHF-Reader, Chip-Card-Reader, und Multi-IO-Port Können erst nach Einbau mit Messgeräten und Produktiv-

	Software auf Funktionalität getestet werden.
Wie wurde damit in der Vergangenheit umgegangen?	Testen der Hardware nach Einbau
Wieso besteht Handlungsbedarf?	Vereinfachung und Optimierung, um Arbeitsstunden der Werkstattmitarbeiter zu verringern.
Wie groß ist der geschätzte Aufwand für die Einarbeitung und Vorbereitung als Entscheidungsgrundlage?	Zwei Wochen

Zielsetzung

Anforderung / Beschreibung	Parameter
Was genau soll am Ende des Projekts entstanden sein?	Eine Software, zum Prüfen der Funktionalität der Hardwarekomponenten
Woran wird der Erfolg im Einzelnen gemessen?	Verständlich für Werkstattmitarbeiter Zeitersparnis und Qualität.
Welche Messverfahren kommen zum Einsatz?	Korrektheit der Testergebnisse durch die Software
Was muss passieren, damit die Lösung realisiert werden kann?	Erstellte DLLs der einzelnen Hardwarekomponenten in C#
Welche Termine gelten?	Fertigstellung Software 17.12.2024
Welches Budget liegt zugrunde?	1500€
Läuft es auf eine Ausschreibung hinaus?	Nein

Produkteinsatz

Anforderung / Beschreibung	Parameter
Unter welchen Rahmenbedingungen soll das Produkt zum Einsatz kommen?	Werkstatt Thermotex

(z.B. Temperatur, klimatische Bedingungen, Druck, Umfeld,…)	
Von wem soll das Produkt bedient werden?	Werkstattmitarbeiter Thermotex
Welche Schnittstellen (Material bzw. Daten) soll das Produkt bieten?	TCP/IP, COM
Welche Bedingungen gelten für die EDV? (z.B. Hardware, Software, Betriebssystem)	Volle Unterstützung auf Windows 10

Funktionale Anforderungen

Anforderung / Beschreibung	Parameter
Welche Funktionen sollen vorhanden sein?	Testen aller Funktionen der Hardwarekomponenten
Was soll das Produkt können oder leisten?	Anzeigen der Testergebnisse der Tests der Hardwarekomponenten
Was soll das Produkt nicht können oder leisten?	Keine Erweiterbarkeit

Nichtfunktionale Anforderungen

Anforderung / Beschreibung	Parameter
Welche Anforderungen werden an die Bedienbarkeit gestellt?	Einfach und schnell zu bedienen
Soll das Produkt erweiterbar sein?	Nein
Sollen Änderungen möglich sein, wenn ja wie genau?	Nein
Soll das Produkt bestimmte Standzeiten erfüllen?	Nein
Welche Zuverlässigkeit muss gegeben sein?	Volle Zuverlässigkeit
Welche Toleranzen werden akzeptiert?	Keine

Welche Ansprüche werden an die Effizienz gestellt?	Steigerung der Effizienz beim Testen der Funktionalität der Hardwarekomponenten

Lieferumfang

Anforderung / Beschreibung	Parameter
Was genau soll in welcher Form geliefert werden?	Software für Werkstatt
Welche Dokumentation soll geliefert werden?	Bedienungsanleitung für Mitarbeiter
Was gehört nicht mehr zum Lieferumfang?	Hardware

Projektphasen

Anforderung / Beschreibung	Parameter
Welche Phasen können berechnet werden?	Alle Phasen

Abnahmekriterien und Qualitätsanforderungen

Anforderung / Beschreibung	Parameter
Welche Qualitätsanforderungen werden an das Projekt gestellt?	Funktionalität der Software

9.3 Pflichtenheft

THERMOTEX	**Pflichtenheft**	**Projekt-Nr:** 12/2024

Projekt-Bezeichnung:	Hardware-TestTool				
Ersteller:	ERE/KDI	Stand:	2	Datum:	03.12.2024

Freigabe von Unternehmensbereich:	Thermotex: SW-EW, Werkstatt, Maschinen-EW	Name / Unterschrift:	Basel Hlal	Datum:	03.12.2024

Die einzelnen Punkte müssen je nach Produkt / Projekt angepasst oder ergänzt werden!

	Anforderungen	Nr.	Beschreibung	Klassifizierung: F / W / U[1]
1	Funktionen		Web-IO Digital	F
		1.1	PC mit Wut Verbunden	F
		1.2	Den Status von Ein- und Ausgängen präsentieren	F
		1.3	Kontrollieren der Ausgänge	F
		1.4	entsprechende Meldungen präsentieren, wenn Wut Events Schicket	F
			UHF-Reader (Sil-9400-MUX4)	
		1.4	PC mit UHF-Reader durch COM verbunden	F
		1.5	PC mit UHF-Reader durch IP-Adresse verbunden	F
		1.6	UHF-Chips Lesen	F
		1.7	Leak Test machen	F
		1.8	entsprechende Meldungen präsentieren, wenn UHF_Reader Events Schicket	F
			Chip-Reader (TS-HR38) //2 Stucke	
		1.9	PC mit Chip-Reader durch COM verbunden	F

[1] F = Forderung, W = Wunsch, U = unklar

		1.10	entsprechende Meldungen präsentieren, wenn Chip-Reader Events Schicket	F
			Datenbank (SQLight benutzen)	
		1.11	Speicher Infos in Datenbank	F
		1.12	Datenbank lesen / präsentieren	F
2	Bedienbarkeit	2.1	Verwendung großer Schriftarten und Schaltflächen, da der Monitor klein ist (22 cm x 15 cm)	F
		2.2	Verwenden farbwechselnde Elemente, um Änderungen einfach und aus der Ferne zu erkennen.	F
		2.3	Um Platz zu sparen, verwenden keine Titelleiste	F
3	Erweiterbarkeit	3.1	mit einer einfachen Bildschirmtastatur, da die Software auf einem Touchscreen-Computer läuft	F
		3.2	automatische Datenbankaktualisierung	F