Hochschule Karlsruhe
Technik und Wirtschaft
UNIVERSITY OF APPLIED SCIENCES

Fakultät Elektro- und Informationstechnik
Studiengang Elektrotechnik – Elektro- und Informationstechnik

Masterthesis

von

David Erb

Referent:	Prof. Dr. Marianne Katz
Korreferent:	Prof. Dr. rer. nat. Klaus Wolfrum
Arbeitsplatz:	
Betreuer am Arbeitsplatz:	
Zeitraum:	18.04.2017 – 18.10.2017

Eidesstattliche Erklärung

Hiermit versichere ich, die vorliegende Masterthesis ohne unzulässige fremde Hilfe selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt zu haben.

Stuttgart, den 11. Mai 2017

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Motivaton

configurAIDER wurde im Rahmen von verschiedenen studentischen Arbeiten erstellt. Aktuell ergeben sich folgende Probleme:

- nicht testbar, da das Sollverhalten nicht immer bekannt ist
- nicht wartbar, da sich keiner im Code auskennt und keine saubere Dokumentation vorhanden ist
- schwer erweiterbar, da sich keiner im Code auskennt und keine saubere Dokumentation vorhanden ist

1.2 Aufgabenstellung

Kapitel 2

Grundlagen

2.1 .NET-Framework

2.2 Windows Presentation Framework(WPF)

2.3 GIGABOX

GIGABOX ist eine Produktfamilie von Steuergeräten der Firma GIGATRONIK. GIGABOX-Steuergeräte werden hauptsächlich in der Fahrzeugentwicklung eingesetzt, um prototypische Kommunikationslösungen im Fahrzeug schnell realisieren zu können. Beispielhafte Anwendungen sind der Einsatz als Gateway zur Übersetzung zwischen unterschiedlichen Busprotokollen und zur Ansteuerung von Treiber-ICs im Automobil.

GIGABOX-Steuergeräte werden in unterschiedlicher Hardwareausstattung und Funktionalität angeboten. Tabelle XX bietet eine Übersicht über die verschiedenen GIGABOX Varianten.

TODO: Welche GIGABOX-Varianten gibt es? Auf Homepage sind nur wenige gelistet. Warum? In Confluence: S,FR Degraded, FR Extended, FR Gateway, FR Gateway BLE, XL Standard -> Vermutung: Nur die auf der Homepage angebotenen werden noch verkauft: BEO, flex-i-FlexRay USB Interface, FlexRay Active Star, Gateway, XL. Warum heißt sense ironCUBE auch GIGABOX? -> ist kein Steuergerät

Eine GIGABOX besteht aus einer Basisplatine und einer Applikationsplatine. Das GIGABOX gate Grundmodul verfügt über Stromversorgung, USB-Anschluss und einen leistungsfähigen Mikrocontroller mit einer Vielzahl von

Standard-Schnittstellen (FlexRay, CAN, LIN) sowie der dazugehörenden Treiber-Software.

TODO: Wenn auf Basisplatine LIN,CAN,Flexray ist, warum haben alle GIGABOXEN diese Bussysteme? Sind grundsätzlich alle Module für LIN,CAN,Flexray vorhanden, aber je nach Applikationsplatine werden nur einzelne Module zur Verfügung gestellt?

Beschreibung Funktionsweise GIGABOX aus Softwaresicht. Skript-Applikation und Bootapplikation. Virtuelle Maschine läuft in Skript-Applikation. Auf virtueller Maschine wird Bytecode ausgeführt. Der Bytecode wurde vom PAWN-Compiler (in configurAIDER) aus einer Skriptdatei erzeugt. Quelle: Bachelorarbeit Christian Eissler Absch. 4.1 (Arbeit über GIGABOX FD -> Funktionsweise bei den alten GIGABOXEN gleich?)

Prozessor Flash-Speicher

Der integrierte Mikrocontroller bietet eine UART-Schnittstelle. Mithilfe eines USB-to-Serial Port Adapters (Korrekt?) kann die GIGABOX per Kabel mit der USB-Schnittstelle eines PCs verbunden werden, um z.B. neu zu flashen.

2.4 Skriptsprache PAWN

Siehe Pawn_Language_Guide.pdf, Foreword
Pawn_Implementation_Guide.pdf, The Compiler, The abstract machine

Warum wurde PAWN gewählt? -frei verfügbar

Vorteile PAWN auf VM gegenüber C? - einfacher zu programmieren -durch VM ist Code hardwareunabhängig und muss nicht an eingesetzten Mikrocontroller angepasst werden -> VM muss aber an Hardware angepasst werden

2.5 CAN

2.6 LIN

2.7 configurAIDER

Der configurAIDER ist eine von GIGATRONIK entwickelte integrierte Entwicklungsumgebung (IDE), die es Benutzern ermöglicht, mit einer GIGABOX

zu kommunizieren und diese zu programmieren. Es können Diagnoseinformationen abgerufen werden und Bytecode auf das Steuergerät geflasht werden. Die IDE besteht aus Editoren (skriptEDITOR und multiEDITOR), Interpreter und Linker. Der scriptEDITOR ist ein Texteditor, mit dem PAWN-Skripte geschrieben werden können. Mit dem multiEDITOR können tabellarisch WENN DANN Anweisungen konfiguriert werden, aus denen ein Skript mit PAWN-Code generiert werden kann. Der multiEDITOR stellt für Benutzer ohne Programmierkenntnisse eine Möglichkeit dar, die GIGABOX mit rudimentären Funktionen zu konfigurieren. Der integrierte Interpreter kann Bytecode aus den erstellten PAWN-Skripten erstellen und auf die GIGABOXEN flashen.

Kapitel 3

Evaluierung Ist-Zustand des configurAIDERs

3.1 Übersicht

Der configurAIDER ist eine von GIGATRONIK entwickelte integrierte Entwicklungsumgebung (IDE), die es Benutzern ermöglicht, mit einer GIGABOX zu kommunizieren und diese zu programmieren. Im Folgenden wird auf die Funktionalität des configurAIDERs eingegangen und die Benutzeroberfläche beschrieben. Im Anschluss wird die Funktionalität und die Usability bewertet. Abbildung 3.1 veranschaulicht die Module, die im configureAIDER enthalten sind.

3.2 Elemente des configurAIDER

3.2.1 Rahmenfenster

Der configurAIDER besteht aus einem Rahmenfenster, in das weitere Fenster eingebettet werden können. Das Rahmenfenster besitzt im oberen Fensterbereich eine Menüleiste in horizontaler Ausrichtung und ist in einem Hintergrund gehalten, der Firmenlogo und -farbe zeigt. Die Menüeinträge ändern sind dynamisch in Abhängigkeit der vom Benutzer geöffneten Fenster. Beim Start des confiurAIDERs ist das Fenster „Verfügbare Geräte“ standardmäßig in das Rahmenfenster eingebettet. Eingebettete Fenster können nicht über die Grenzen des Rahmenfensters hinausgezogen werden.

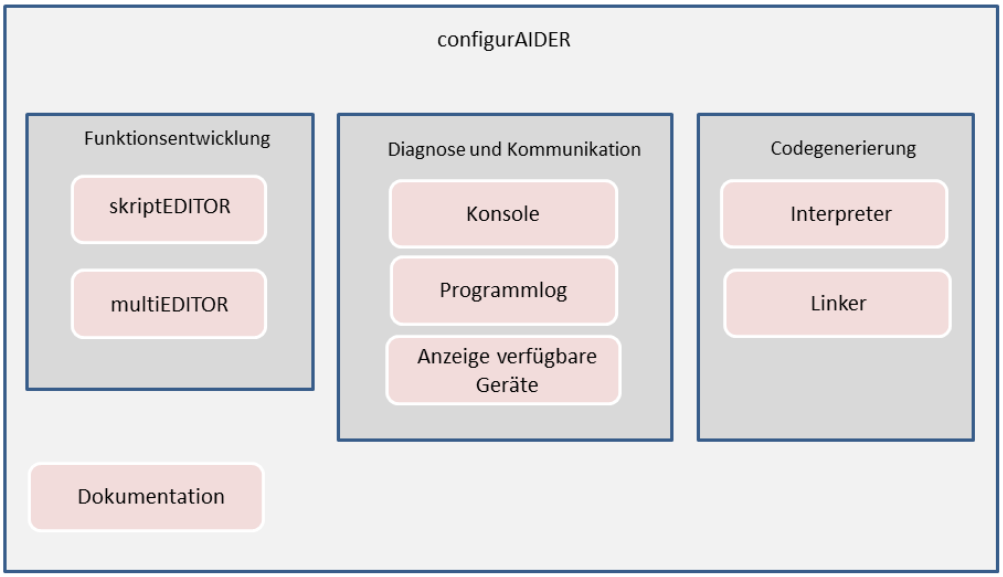

Abbildung 3.1 – Übersicht configurAIDER

(Bild einfügen)

3.2.2 Fenster „Verfügbare Geräte“

Im Fenster „Verfügbare Geräte“ werden alle GIGABOXen angezeigt, die an der USB-Schnittstelle des PC angeschlossen sind. Es wird der Modellname, die Seriennummer und der Verbindungsstatus von GIGABOXen ausgegeben, die am PC erkannt wurden. Über dieses Fenster kann eine GIGABOX angewählt werden, mit der eine Kommunikation gestartet werden soll. Wenn keine reale GIGABOX zur Verfügung steht, kann eine GIGABOX simuliert werden. Die simulierte GIGABOX wird wie eine reale GIGABOX gelistet.

Nach der Anwahl eines GIGABOX-Modelles wird versucht, eine Verbindung zu dem Gerät herzustellen (wird das tatsächlich hardwaremäßig versucht oder ist die Anzeige nur dazu da, dem Benutzer zu signalisieren, welches Gerät er konfiguriert). Bei erfolgreicher Verbindungsherstellung wird das Gerät grün hinterlegt und es öffnet sich ein Fenster zur Editorauswahl. Dort kann ausgewählt werden, ob die Konsole, der scriptEDITOR oder der multiEDITOR geöffnet werden soll. Bei einigen GIGABOX-Modellen steht kein multiEDITOR zur Verfügung. (Bild des Fensters mit Geräten einfügen)

3.2.3 Fenster „Konsole“

Über die Konsole können direkt Befehle an verbundene Geräte abgegeben werden. Sie kann z.B eingesetzt werden zur Abfrage von Diagnoseinformationen und um einen Reset zu veranlassen.

3.2.4 Fenster „Programmlog“

Das „Programmlog“-Fenster enthält verschiedene Informationen über den Zustand sowie die letzten Aktionen des configurAIDERs. Sollte es zu Fehlern im Programmablauf kommen, werden diese hier geloggt und informieren den Benutzer über die aufgetretenen Fehler.

3.2.5 Fenster „scriptEDITOR“

Der scriptEDITOR ist ein Texteditor zum Erstellen und Editieren von Funktionen in PAWN. Um den scriptEDITOR aufzurufen, muss zuerst eine Verbindung zu einer realen oder simulierten GIGABOX hergestellt werden über das

Fenster „Verfügbare Geräte“. Anschließend öffnet sich automatisch ein Fenster zur Editorauswahl. Dort kann der scriptEDITOR angewählt werden. Mit dem scriptEDITOR erstellte Skriptdateien haben die Dateiendung .gt**.p. Die Sterne sind zu ersetzen mit einem Kürzel, das abhängig ist vom verbundenen GIGABOX-Modell. Für die unterschiedlichen GIGABOX-Modelle werden also Skripte mit verschiedenen Dateiendungen erstellt, da die Modelle unterschiedliche Funktionen implementieren. Bsp: GIGABOX gate FR extended: .gtfre.p GIGABOX gate XL: .gtxl.p GIGABOX gate gateway: .gtfrg.p

In der obersten Fensterzeile ist eine Buttonleiste angeordnet mit den folgenden Buttons:

- Neu: Öffnet ein neues Skript
- Öffnen: Öffnet ein vorhandenes Skript
- Speichern: Speichert das geöffnete Skript unter einem bereits festgelegt Dateinamen und Pfad
- Speichern unter: Speichert das geöffnete Skript unter einem anzugebenden Dateinamen und Pfad
- Kompilieren: Interpreter erzeugt Bytecode aus dem PAWN-Quellcode
- Herunterladen: Interpreter erzeugt Bytecode aus dem PAWN-Quellcode. Die PAWN-Skriptdatei wird als ZIP-Datei komprimiert. Anschließend wird der Bytecode und das gezippte Skript auf die virtuelle Maschine der GIGABOX übertragen.
- Heraufladen: Die als ZIP-Datei komprimierte Skriptdatei auf dem Steuergerät wird auf den PC übertragen, entzippt und im Texteditor geöffnet.
- Reset: Führt einen Reset der GIGABOX durch
- Log: Öffnet Konsole

Mittig ist der Texteditor angeordnet. Hier kann ein Skript angezeigt und bearbeitet werden. Das gleichzeitige Anzeigen von mehreren Skripten, die als Tabs oder ähnliches angeordnet sind, ist nicht möglich. Es stehen bei der Entwicklung typische Unterstützungswerkzeuge wie Codefolding zur Verfügung. (Aufzählung der unterstützenden Werkzeuge wie Autovervollständigung etc.)

Unten befindet sich ein Fenster zur Ausgabe von Fehlern, Warnungen und Erfolgsmeldungen.

Bild einfügen

3.2.6 Fenster „multiEDITOR“

Der „multiEDITOR“ ist ein Editor zur Funktionskonfiguration mit WENN-DANN-Anweisungen in Tabellen. Er soll es Benutzern mit geringer Programmiererfahrung ermöglichen, einfache Funktionen für die GIGABOX zu entwickeln. Aus den konfigurierten Funktionstabellen kann anschließend automatisch ein PAWN-Skript generiert werden. Das erstellte PAWN-Skript kann dann interpretiert und der Bytecode auf die virtuelle Maschine der GIGABOX aufgespielt werden.

In der obersten Fensterzeile ist eine Buttonleiste angeordnet mit den folgenden Buttons:

- Neu: Öffnet ein neues Skript
- Öffnen: Öffnet ein vorhandenes Skript
- Speichern: Speichert das geöffnete Skript unter einem bereits festgelegt Dateinamen und Pfad
- Speichern unter: Speichert das geöffnete Skript unter einem anzugebenden Dateinamen und Pfad
- Generieren: Aus den mittels Tabelle konfigurierten Funktionen wird ein PAWN-Skript generiert
- Kompilieren: Interpreter erzeugt Bytecode aus dem PAWN-Quellcode
- Herunterladen: Aus den mittels Tabelle konfigurierten Funktionen wird ein PAWN-Skript generiert. Interpreter erzeugt Bytecode aus dem PAWN-Quellcode. Die PAWN-Skriptdatei wird als ZIP-Datei komprimiert. Anschließend wird der Bytecode und das gezippte Skript auf die virtuelle Maschine der GIGABOX übertragen.
- Heraufladen: Die als ZIP-Datei komprimierte Skriptdatei auf dem Steuergerät wird auf den PC übertragen und entzippt. PAWN-Code wird rückgewandelt in Tabelle mit WENN-DANN-SONST-Anweisungen. Die funktioniert nur, wenn der von der GIGABOX hochgeladenen Code ursprünglich mithilfe des multiEDITORs erzeugt wurde.
- DBC: Es können Vector-DBC Dateien oder AUTOSAR .arxml Dateien erstellt oder geöffnet werden. Dabei handelt es sich um Datenbanken, in denen CAN-Botschaften und Signale definiert sind
- Reset: Führt einen Reset der GIGABOX durch

- Log: Öffnet Konsole

Unter der Buttonleiste befindet sich eine Tabelle, in der CAN-Botschaften und Signale erstellt und bearbeitet werden können. Botschaften und Signale, die aus Datenbank-Dateien (.dbc oder .arxml) geladen wurden, werden hier angezeigt.

Darunter befindet sich eine Tabelle zur Konfiguration von Funktionen mithilfe von WENN-DANN-SONST-Anweisungen. Für jede Funktion wird eine neue Tabelle erstellt. Jede Tabelle besteht mindestens aus 3 Zeilen mit den Anweisungen WENN, DANN und SONST. Es können mehrere WENN-Anweisungszeilen erstellt werden, die mit dem Bedingungsoperator UND oder ODER verknüpft werden müssen. Außerdem können mehrere DANN und SONST-Anweisungen erstellt werden, die jeweils miteinander über den UND-Operator verknüpft werden. Unter jeweils jede DANN und SONST-Anweisung kann eine neue, untergeordnete WENN-DANN-SONST-Anweisung eingefügt werden. Damit sind verschachtelte Anweisungen möglich.

In den Spalten der WENN-Anweisungszeile können digitale und analoge Eingänge, digitale Ausgänge sowie eingehende Busbotschaften auf definierte Werte geprüft werden. Wenn die dort definierten Werte angenommen werden, wird die DANN-Anweisungenszeile ausgeführt. Hier können digitale Ausgänge durchgeschaltet oder ausgeschaltet werden. Das Absetzen von Busbotschaften ist nicht möglich.

Wenn die in der WENN-Anweisungszeile definierten Werte nicht angenommen werden, wird die SONST-Anweisungszeile ausgeführt. Auch hier können digitale Ausgänge durchgeschaltet oder ausgeschaltet werden. Das Absetzen von Busbotschaften ist nicht möglich.

Bild einfügen

3.3 Quellcode des configurAIDERs

Was wurde mit WinForms gemacht, was mit WPF?

Entstand aus mehreren studentischen Arbeiten

Diagramm mit bestehender Architektur erstellen

3.4 Beurteilung

Gliederung: Trennung zwischen Funktionalität und Usability sinnvoll?

3.4.1 Funktionalität

Der configurAIDER enthält alle grundlegend notwendigen Funktionen, die eine integrierte Entwicklungsumgebung für Steuergeräte benötigt. Das Tool erkennt GIGABOXEN, die mit dem PC verbunden sind und ermöglicht die Kommunikation mit diesen. Dazu bietet es Editoren zur Erstellung von PAWN-Code und einen integrierten PAWN-Interpreter, mit dem Bytecode erzeugt werden kann. Der erzeugte Bytecode kann dann auf den Flash-Speicher einer GIGABOX übertragen werden.

Verfügbare Geräte

Die Simulation von Geräten führt gelegentlich zu Fehlverhalten. Beim Versuch einer Verbindungsherstellung zu einer simulierten GIGABOX wird dabei ein Fehler über das Programmlog-Fenster ausgegeben (ERROR: Internal device table obscured!). Bei nochmaligem Versuch kann die Verbindung erfolgreich hergestellt werden.

scriptEDITOR

Der scriptEDITOR bietet eine akzeptable funktionierende Umgebung, um PAWN-Code zu entwickeln. Er kann punktuell um einige Features verbessert werden, um dem Entwickler ein angenehmeres Programmieren zu ermöglichen (siehe Liste Verbesserungen). Bsp: Den Entwickler unterstützende Features wie Autovervollständigung sind teilweise vorhanden. Allerdings gibt es dort noch Potenzial zur Verbesserung der bereits implementierten Features und zur Integration von neuen Features. -> sind alles Features die die Usability verbessern

Zu integrieren: Anzeige der Zustände von Ein- und Ausgängen

Störend ist die fehlende Möglichkeit, mehrere Skripte parallel öffnen und bearbeiten zu können. Dies würde es dem Benutzer beispielsweise erleichtern, Code aus einem fertigen Skript in ein neues Skript reinzukopieren um bereits entwickelten Code wiederzuverwenden.

Es sollte die Möglichkeit geben, aus einer Datenbank (.dbc und .arxml) ein include-File mit allen Signalen (Name, ID, Inhalt) zu erzeugen. (So ein File wird bereits benutzt und von Olli eingebunden. Wie wird es bisher erzeugt?)

multiEDITOR

Hinter dem multiEDITOR steckt grundsätzlich eine interessante Idee. Es soll Kunden mit wenig Programmierkenntnissen eine Möglichkeit geboten werden, GIGABOXen selbständig zu programmieren. Nach Analyse von entwickelten PAWN-Skripten wurde allerdings festgestellt, dass der multiEDITOR von der Funktionsfähigkeit zu rudimentäre Möglichkeiten für die meisten Anwendungen bietet. Die Tatsache, dass keine Busbotschaften abgesetzt werden können, schränkt die Einsatzmöglichkeiten des multiEDITORs sehr stark ein.

Reaktion auf LIN-Nachrichten nicht möglich (?)

Da keine switch-case-Anweisung zur Verfügung steht, müssen tief verschachtelte WENN-DANN-SONST-Anweisungen erstellt werden, um gleiches Verhalten zu erreichen. Die Anweisungstabelle wird dadurch sehr komplex und unübersichtlich.

Desweiteren ist es von Unternehmensseite eher unerwünscht, dass Kunden die gekauften GIGABOXEN selbstständig programmieren. Wenn das Unternehmen von Kunden den Auftrag bekommt, die gewünschten Steuergerätefunktionen zu implementieren, kann neben dem Verkauf der GIGABOX zusätzlich für die Programmierdienstleistung Geld generiert werden.

3.4.2 Usability

Einen Anhaltspunkt zu Bewertung der Usability einer Software bietet die EN ISO 9241-110. Hier werden die Grundsätze der Dialoggestaltung beschrieben.

- Aufgabenangemessenheit: „Ein Dialog ist aufgabenangemessen, wenn er den Benutzer unterstützt, seine Arbeitsaufgabe effektiv und effizient zu erledigen.“
- Selbstbeschreibungsfähigkeit: „Ein Dialog ist in dem Maße selbstbeschreibungsfähig, in dem für den Benutzer zu jeder Zeit offensichtlich ist, in welchem Dialog, an welcher Stelle er sich befindet, welche Handlungen unternommen werden können und wie diese ausgeführt werden können.“
- Lernförderlichkeit

- Steuerbarkeit
- Erwartungskonformität: „Ein Dialog ist erwartungskonform, wenn er konsistent ist und den Merkmalen des Benutzers entspricht, z.B. seinen Kenntnissen aus dem Arbeitsgebiet, seiner Ausbildung und seiner Erfahrung sowie den allgemein anerkannten Konventionen.“
- Individualisierbarkeit
- Fehlertolerant: „Ein Dialog ist fehlertolerant, wenn das beabsichtigte Arbeitsergebnis trotz erkennbar fehlerhafter Eingaben entweder mit keinem oder mit minimalem Korrekturaufwand durch den Benutzer erreicht werden kann.“

Quelle: Wikipedia -> überprüfen ob andere Quelle verfügbar ist, am Besten die Norm direkt

Übersichtsbild von Dr. Wettach, Prozessvisualisierung einfügen

Das Rahmenfenster enthält rechts oben Buttons zur Fenstersteuerung (Minimieren, Maximieren, Schließen) wie es Standard ist bei Desktop-Software. Wenn im Rahmenfenster ein zusätzliches Fenster geöffnet und maximiert wird (z.B. Skripteditor), ist schwer erkennbar, dass es sich um zwei verschiedene Fenster handelt. Für das eingebettete Fenster werden zusätzlich Buttons zur Fenstersteuerung (Minimieren, Maximieren, Schließen) rechts oben neben der Menüleiste eingefügt. Es kommt durch den Benutzer leicht zu Fehlbedienung, wenn das eingebettete Fenster geschlossen werden soll, der Benutzer aber das Rahmenfenster schließt. Verstößt gegen Selbstbeschreibungsfähigkeit

scriptEDITOR

Keine Möglichkeit, letzte Aktion rückgängig zu machen -> schlechte Steuerbarkeit

Keine Erkennung von Syntaxfehlern -> schlechte Selbstbeschreibungsfähigkeit

Buttons Herunterladen/Heraufladen sind missverständlich -> schlechte Selbstbeschreibungsfähigkeit

Nur ein Skript kann geöffnet werden, nicht mehrere gleichzeitig -> Aufgabenangemessenheit

Autovervollständigung nur teilweise umgesetzt -> Aufgabenangemessenheit

Codeblöcke können nicht auskommentiert werden über Buttons/Tastenkombination -> Aufgabenangemessenheit

2 Anzeigen für Fehler-/Erfolgsmeldungen -> (Aufgabenangemessenheit), keine genaue Zuordnung möglich

multiEDITOR

Im Signaleditor ist Bit/CAN-Bit Unterschied nicht klar ersichtlich -> Selbstbeschreibungsfähigkeit

3.4.3 Ergebnis

Kapitel 4

Anforderungsanalyse

4.1 Ermittlung der Anforderungen

Im Folgenden soll eine vollständige Anforderungsanalyse für eine Entwicklungsumgebung für GIGABOX-Steuergeräte durchgeführt werden.

4.1.1 Funktionale Anforderungen

- Erstellung von Projekten mit Baumstruktur
  - Innerhalb eines Projektes können verschiedene GIGABOX-Modelle angelegt werden
  - Für jedes angelegte GIGABOX-Modell können PAWN-Skripte erstellt werden mit der zum Modell passenden Dateiendung .gt**.p
  - Für jedes angelegte GIGABOX-Modell können Include-Files hinzugefügt werden
- Texteditor zur Erstellung von PAWN-Skripten
  - mehrere Skripte lassen sich parallel über Tabs öffnen
  - Copy/Paste
  - Suchen/Ersetzen
  - Möglichkeit, letzte Schritte rückgängig zu machen
  - Anzeige aller implementierten Funktionen zur Navigation im Skript
  - Anzeige aller instanziierten Variablen

  - (Signale aus .dbc/.arxml-Dateien werden automatisch als PAWN-Variablen instanziiert)
  - Routing-Editor
    * Laden und Anzeigen von Botschaften/Signalen aus Datenbank (.dbc/.arxml)
    * Eigene Botschaft kann definiert werden mit ID sowie Channel, auf die sie gelegt werden soll
    * Inhalt der eigenen Botschaft kann aus Signalen, die in Datenbank definiert sind, zusammengestellt werden
    * Aus der im Routing-Editor zusammengestellten Botschaft kann PAWN-Code generiert werden und in die Funktion OnCanRxEvent() eines geöffneten Skriptes im Texteditor eingefügt werden
  - Bei der Codeentwicklung unterstützende Funktionen. Ziel: Schnelles und effektives Schreiben von fehlerfreiem Code

- Frei andockbare Fenster
- Erkennung von GIGABOXEN, die mit dem PC verbunden sind
  - Anzeige der verfügbaren GIGABOXEN
  - Verbindung zu einer GIGABOX herstellen
- PAWN-Compiler
  - Interpreter zum Übersetzen von PAWN-Code in Bytecode
  - Linker, um Bytecode zusammenzufügen
- Beschreiben/Auslesen des Flash-Speichers der GIGABOX
  - Übertragen des Bytecodes vom PC auf den Flash-Speicher der GIGABOX
  - Auslesen des auf einer GIGABOX aufgespielten Bytecodes
- Konsole
  - Eingabe von Befehlen
  - Ausgabe von Meldungen/Informationen
- Steuern und Beobachten von Ein-/Ausgängen und Timer

  - Steuern und Beobachten von DIN, AIN, DOUT, SWITCH
  - CAN-/LIN-Botschaften senden (einmalig und zyklisch)
  - CAN/LIN beobachten über Schnittstelle zu CANoe
  - Steuern und Beobachten der internen Timer

- Debugging
  - Setzen von Breakpoints
  - Zeilenweise Steuerung der Anweisungsausführung

- Skriptgenerierung aus MATLAB/Simulink
  - Modellbasierte Entwicklung von Regelalgorithmen in MATLAB/Simulink. Generierung eines PAWN-Skriptes aus Simulink-Modell.

- Bereitstellung von Dokumentationen
  - Dokumentation zur GIGABOX
  - Dokumentation zum condigurAIDER
  - Dokumentation zu PAWN

Das Use-Case-Diagramm in Abbildung 4.1 veranschaulicht die grundlegenden Funktionen, die an das Tool gestellt werden.

4.1.2 Nichtfunktionale Anforderungen

Entwicklung des Tools mit WPF unter Verwendung MVVM-Pattern

Entwicklung für Windows 32bit + 64bit als Zielbetriebssystem

Entwicklung nach V-Modell

Qualitätskriterien an Software nach ISO 9126 (aus Wiki)

- Wartbarkeit
  - Analysierbarkeit:
    Coding-Richtlinien sollen eingehalten werden: Verständliche Kommentare, einheitliche Namensgebung etc

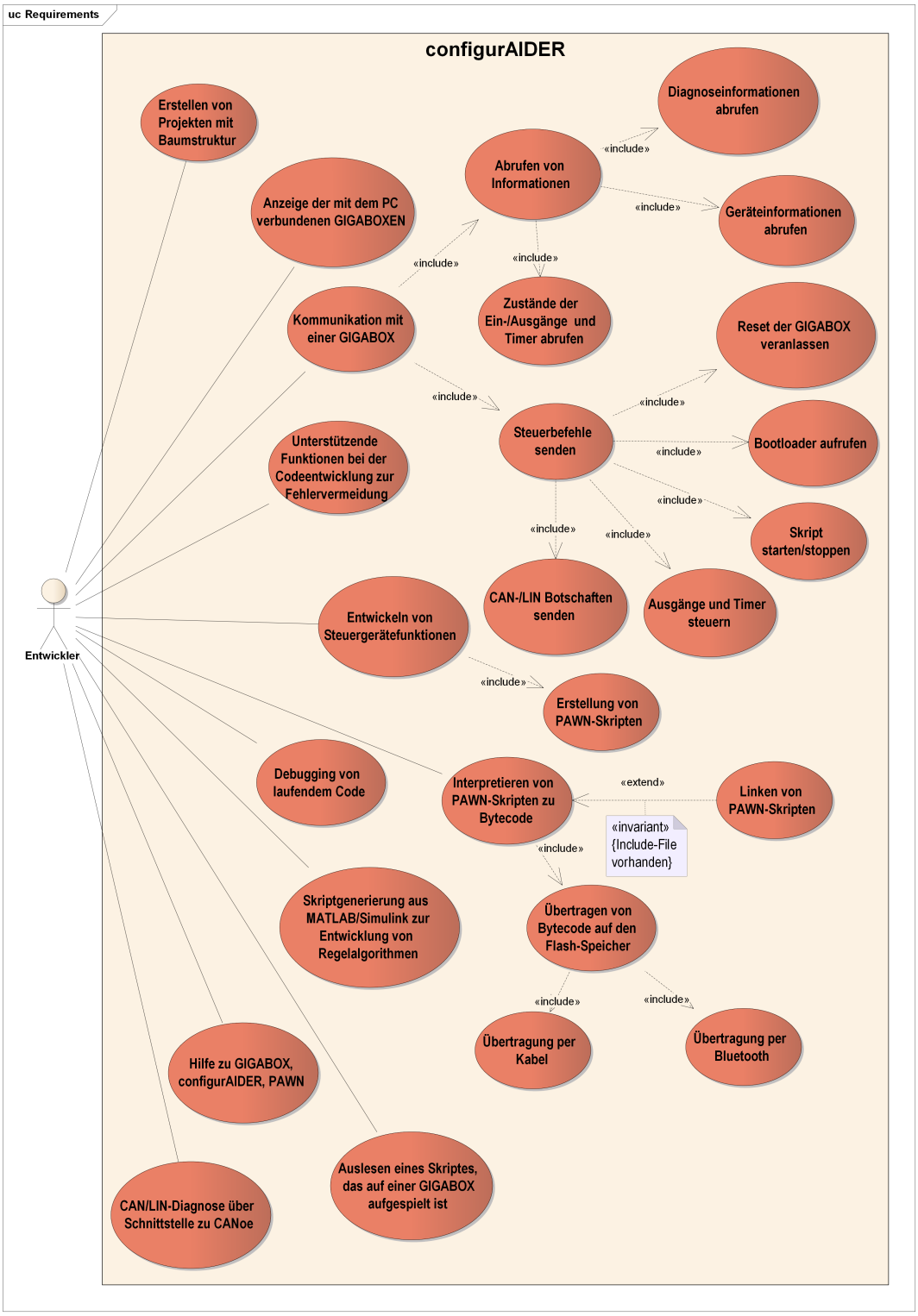

Abbildung 4.1 – Use-Case-Diagramm funktionale Anforderungen

  - Modifizierbarkeit:
    SOLID-Prinzipien einhalten, um möglichst wenig Abhängigkeiten unter den Klassen zu erreichen. Dadurch treten bei Modifikationen innerhalb einer Klasse weniger schwer vorhersehbare Fehler auf
  - Testbarkeit:
    Keine Ahnung auf was bei der Entwicklung geachtet werden soll um gute Testbarkeit zu erreichen

- Benutzbarkeit (genauer definiert in Norm EN ISO 9241-110)
- Effizienz
  - Akzeptable Zeit bis Tool gestartet ist und verwendbar ist
  - Unmittelbare und flüssige Reaktion auf Benutzereingaben
- Übertragbarkeit
  - Anpassbarkeit: Möglichkeit, Software an verschiedene Umgebungen anzupassen
    Verschiedene Betriebssysteme? 32/64Bit?
- Zuverlässigkeit
  - Reife: Geringe Versagenshäufigkeit durch Fehlerzustände: Ausreichendes Testing
  - Wiederherstellbarkeit: Fähigkeit, bei einem Versagen das Leistungsniveau wiederherzustellen und die direkt betroffenen Daten wiederzugewinnen. Zu berücksichtigen sind die dafür benötigte Zeit und der benötigte Aufwand.

4.2 Bewertung und Klassifizierung der Anforderungen

4.2.1 Funktionale Anforderungen

- Erstellung von Projekten mit Baumstruktur
  - Innerhalb eines Projektes können verschiedene GIGABOX-Modelle angelegt werden (++)

  - Für jedes angelegte GIGABOX-Modell können PAWN-Skripte erstellt werden mit der zum Modell passenden Dateiendung .gt**.p (++)
  - Für jedes angelegte GIGABOX-Modell können Include-Files hinzugefügt werden (+)

- Texteditor zur Erstellung von PAWN-Skripten
  - mehrere Skripte lassen sich parallel über Tabs öffnen (++)
  - Copy/Paste (++)
  - Suchen/Ersetzen (++)
  - Möglichkeit, letzte Schritte rückgängig zu machen (++)
  - Anzeige aller implementierten Funktionen zur Navigation im Skript (+)
  - Anzeige aller instanziierten Variablen (0)
  - (Signale aus .dbc/.arxml-Dateien werden automatisch als PAWN-Variablen instanziiert)
  - Routing-Editor (+)
    * Laden und Anzeigen von Botschaften/Signalen aus Datenbank (.dbc/.arxml)
    * Eigene Botschaft kann definiert werden mit ID sowie Channel, auf die sie gelegt werden soll
    * Inhalt der eigenen Botschaft kann aus Signalen, die in Datenbank definiert sind, zusammengestellt werden
    * Aus der im Routing-Editor zusammengestellten Botschaft kann PAWN-Code generiert werden und in die Funktion OnCanRxEvent() eines geöffneten Skriptes im Texteditor eingefügt werden
  - Bei der Codeentwicklung unterstützende Funktionen. Ziel: Schnelles und effektives Schreiben von fehlerfreiem Code (++)

- Frei andockbare Fenster (+)

- Erkennung von GIGABOXEN, die mit dem PC verbunden sind
  - Anzeige der verfügbaren GIGABOXEN (++)
  - Verbindung zu einer GIGABOX herstellen (++)

- PAWN-Compiler

  - Interpreter zum Übersetzen von PAWN-Code in Bytecode (++)
  - Linker, um Bytecode zusammenzufügen (++)

- Beschreiben/Auslesen des Flash-Speichers der GIGABOX
  - Übertragen des Bytecodes vom PC auf den Flash-Speicher der GIGABOX (++)
  - Auslesen des auf einer GIGABOX aufgespielten Bytecodes (++)

- Konsole
  - Eingabe von Befehlen (++)
  - Ausgabe von Meldungen/Informationen (++)

- Steuern und Beobachten von Ein-/Ausgängen und Timer
  - Steuern und Beobachten von DIN, AIN, DOUT, SWITCH (+)
  - CAN-/LIN-Botschaften senden (einmalig und zyklisch) (0)
  - CAN/LIN beobachten über Schnittstelle zu CANoe (-)
  - Steuern und Beobachten der internen Timer (0)

- Debugging
  - Setzen von Breakpoints (0)
  - Zeilenweise Steuerung der Anweisungsausführung (0)

- Skriptgenerierung aus MATLAB/Simulink
  - Modellbasierte Entwicklung von Regelalgorithmen in MATLAB/Simulink. Generierung eines PAWN-Skriptes aus Simulink-Modell. (--)

- Bereitstellung von Dokumentationen
  - Dokumentation zur GIGABOX (++)
  - Dokumentation zum condigurAIDER (++)
  - Dokumentation zu PAWN (++)

4.2.2 Nichtfunktionale Anforderungen

Kapitel 5

Konzeption

5.1 Übersicht

Ziel des Kapitels ist es, Konzepte aufzuzeigen, wie die in der Anforderungsanalyse erarbeiteten Anforderungen umgesetzt werden können.

5.2 Erweiterung des bestehenden configurAIDERs um neue Features

Der bestehende Code des configurAIDERs wird erweitert um neue Features, die bei der Bewertung der Anforderungen in Abschnitt 4.2 mit (+) oder (++) bewertet wurden.

Neue Funktionen:

- Projektverwaltung: Es können Projekte angelegt werden, in denen unterschiedliche Modelle der GIGABOX projektiert sind und die zugehörigen Skripte+Include-Files
- Texteditor wird erweitert um:
  1. Möglichkeit, letzte Schritte rückgängig zu machen
  2. Anzeige aller implementierten Funktionen zur Navigation im Skript
  3. Bei der Codeentwicklung unterstützende Funktionen
  4. Routing-Editor

- Frei andockbare Fenster
- Steuern und Beobachten von DIN, AIN, DOUT, SWITCH

Vorteil:

- Das Tool erhält damit einen erhöhten Funktionsumfang, als wichtig bewertete funktionale Anforderungen können umgesetzt werden.

Nachteil:

- Nichtfunktionale Anforderungen wie Qualitätskriterien an Software nach ISO9126 (Wartbarkeit, Usability, Effizienz, Übertragbarkeit, Zuverlässigkeit) können nicht erfüllt werden, sondern werden sich verschlechtern durch eine weitere Funktionserweiterung des Tools
- Viel Einarbeitung in bestehenden Code nötig

5.3 Neuentwicklung des configurAIDERs

Das Tool wird von Grund auf neu entwickelt. Es sollen die funktionalen und nichtfunktionalen Anforderungen aus Kapitel 4 umgesetzt werden. Im Rahmen dieser Arbeit sollen vorerst nur die wichtigsten funktionalen Anforderungen umgesetzt werden (mit ++ bewertet). Allerdings sollen alle nichtfunktionalen Anforderungen erfüllt werden.

Funktionen:

- Projektverwaltung: Es können Projekte angelegt werden, in denen unterschiedliche Modelle der GIGABOX projektiert sind und die zugehörigen Skripte+Include-Files
- Texteditor
  - mehrere Skripte lassen sich parallel über Tabs öffnen
  - Copy/Paste
  - Suchen/Ersetzen
  - Möglichkeit, letzte Schritte rückgängig zu machen
  - Bei der Codeentwicklung unterstützende Funktionen. Ziel: Schnelles und effektives Schreiben von fehlerfreiem Code

- Erkennung von GIGABOXEN, die mit dem PC verbunden sind
- PAWN-Compiler
- Beschreiben/Auslesen des Flash-Speichers der GIGABOX
- Konsole
- Bereitstellen von Dokus

Vorteil:

- Tool wird einfacher erweiterbar als bisheriger Stand. Dies ermöglicht es, zukünftig einfacher neue Features zu integrieren.
- Besser wartbar: Bugs können in Zukunft besser behoben werden, da Verhalten des neuen Codes besser bekannt ist
- Usability kann verbessert werden
- Tool konsistent mit WPF realisiert unter aktueller .NET Version 4.6
- Keine lange Einarbeitung in alten Code nötig

Nachteil:

- In Entwurf der neuen Softwarearchitektur muss Zeit investiert werden
- Tool bietet keinen direkten Mehrwert in Form von höherem Funktionsumfang
- Insgesamt höherer Aufwand um die gestellten funktionalen Anforderungen zu erreichen

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis