Projektdokumentation

**Projekt42**

C++/Qt Implementierung eines Prototypen

für Verkaufsoberflächen

A description...

ETB Electronic Team

Beratungs- und Vertriebs GmbH

Grabauerstrasse 53

21493 Schwarzenbek

Durchführung

Jan Schneidereit

Fachinformatiker / Anwendungsentwicklung

# 1 Projektdefinition

## 1.1 Projektauftrag

Die ETB Electronic Team Beratungs- und Vertriebs GmbH biete eine Vielzahl von Lösungen im Bereich der Telemetrie, sowie Mensch-Maschine-Systeme für Verkaufsautomaten an. Die Aufgabe dieses Projektes beinhaltet die Proof of Concept Implementierung einer neuen POS/Kiosk Oberfläche für Vending Automaten.

Mittels dieser Studie soll nachgewiesen werden, in welchem Umfang sich eine alternative Lösung für die bereits bestehende Verkaufsoberfläche auf kleinere Hard- und Software Systeme portieren lässt.

Das Projekt selbst besteht aus der Kernsoftware, die auf einem Embedded Linux System aufgesetzt wird. Die Kernsoftware selbst wird in ein Backend für Programmlogiken und Kommunikationsschnittstellen, sowie ein Frontend für die Grafische Präsentation von Funktionalität an den Benutzer unterteilt.

Das Backend stellt die Grundlegende Schnittstelle zwischen Grafischer Oberfläche und der Hardware dar. Die Kommunikation mit der Maschine ist nicht Bestandteil dieses Projektes. Über die dafür nötige Implementierung wird erst nach der Beurteilung des Prototypen durch den Auftraggeber entschieden.

Das Frontend soll mindestens 3 verschiedene Oberflächen beinhalten, die die verschiedene Zustände einer Verkaufsoberfläche repräsentieren. Unter anderem eine Produktauswahl, eine Bedienoberfläche zum variieren eines Produktes, sowie ein Begrüßungsschirm. Weitere Oberflächen sind optional und dienen ausschließlich der Vervollständigung des Prototypen.

Diese Oberflächen sollen auf zwei verschiedene Arten implementiert werden können.

Zum einen speziell für diese Software entwickelte Oberflächen, die direkt in einer dafür vorgesehen Sprache geschrieben werden. Zum anderen Dynamisch generierte Oberflächen, die aus dem bestehenden Backend für solche Oberflächen generiert werden. Dies erfordert eine Anpassung des Backends dieser Software an das Format der Oberflächen aus dem bestehenden System.

Die Bedienung des Systems soll wie im Äquivalent auf Touchscreen Bedienung optimiert sein.

## 1.2 Projektumfeld

Das Projekt gehört thematisch zur Abteilung für die Entwicklung von Lösungen rund um Verkaufsautomaten. In diesem Umfeld werden Prototypen und Serienreife Produkte für Großkunden entwickelt. Produkte, die in diesem Umfeld vertrieben werden, sind für den industriellen Einsatz konzipiert, daher werden hier Systeme für den Dauerbetrieb ausgelegt. Da es sich um ein Proof of Concept handelt wird erst nach Projektabschluss über eine Integration des Projektes in die bestehende Entwicklung von POS (Point of Sale) Systemen entschieden.

…

## 1.3 Projektabgrenzung

Die Software, die in diesem Projekt umgesetzt wird, ist zum größten Teil unabhängig von bereits bestehenden Systemen. Die zugrunde liegende Technologie wurde bisher nicht eingesetzt und erfordert eine gänzliche Neukonzipierung. Bereits zur Verfügung gestellte Schnittstellen, sollen im späteren Verlauf ohne große Anpassungen weiter genutzt werden können.

# 2 Projektplanung

### 2.1.1 Ist-Analyse

### 2.1.2 Soll-Konzept

### 2.1.3 Anforderungen

## 2.2 Ressourcenplanung

### 2.2.1 Zeitplanung

### 2.2.2 Kostenplanung

# 3 Realisierung

## 3.1 Die Entwicklungsumgebung

Die Kernsoftware wird in einer C++/Qt basierten Umgebung entwickelt.

Als IDE wird der QtCreator verwendet, einem C++ orientierten Editor, der speziell für die

Verwendung des Qt Frameworks entwickelt wurde. Diese IDE ermöglicht ein sehr effizientes Arbeiten durch zahlreiche Funktionen wie Syntax Highlighting, Code Completion, Semantic Navigation und GUI Designer für sowohl Widget basierte Oberfläche als auch QML basierte Oberfläche.

### 3.1.1 Programmsprache

Die benutzte Programmiersprache für dieses Projekt ist C++.

Der verwendete Compiler unter Windows ist MinGW. Eine Portierung der GCC (GNU Compiler Collection) für Windows Systeme. MinGW wurde in der Version

### 2.1.2 Framework

Als Framework wird das C++ basierte Qt Framework genutzt. Dieses Framework biete umfangreiche Bibliotheken für das Erstellen grafischer Oberflächen, ein intuitives

File-Handling, die Implementierung OpenGL basierter Software, sowie die Anbindung an Netzwerke und vieles mehr. Die grafische Schnittstelle dieses Projektes wird zum größten Teil mit einer neuen Technik namens "Qt Quick" umgesetzt. Diese Technologie beinhaltet eine deklarative Auszeichnungssprache "QML", die es ermöglicht auf sehr einfache Weise grafische Oberflächen zu beschreiben, dadurch sind auch Designer ohne C++ Kenntnisse in der Lage Oberflächen zu gestallten. Außerdem beinhaltet "Qt Quick" die "QtDeclarative" Module. Diese Bibliotheken dienen dazu, die mit QML beschriebenen Oberflächen auf C++ Ebene umzusetzen und zu rendern. Dieses Verfahren führt zu verbesserter Performance was sich gerade auf Target Systemen mit wenigen Ressourcen bezahlt macht. Qt Quick wurde aus diesem Grund von Nokia initialisiert und stark unterstützt, um Mobile Systeme wie Smartphones mit Qt Quick auszustatten.

Das Qt Framework wurde Ursprünglich von Trolltech entwickelt und wurde später von Nokia übernommen. Es steht in verschiedenen Lizenzmodellen zur Verfügung und wird im Rahmen dieses Projektes unter der LGPL Lizenz verwendet.

## 3.2 Schnittstellendefinition

## 3.3 Softwaredesign

## 3.4 Implementierung

### 3.4.1 Backend

### 3.4.2 Frontend

### 3.4.3 Schnittstellen

## 3.5 Konfiguration des Zielsystems

### 3.6 Deployment

### 3.7 Testphase

# 4 Projektbewertung

## 4.1 Funktionalität der Implementierung

## 4.2 Vergleich mit Alternativsystem

## 4.3 ...

# 

# 5 Glossar

**Backend – zu Deutsch „hinteres“ oder „unteres“ Ende. Das Backend beschreibt in diesem Fall die Schnittstelle des System zum darunter liegenden Hardwaresystem.**

**Während das Frontend für die Kommunikation mit dem Benutzer entwickelt wird ist das Backend ausschließlich für die Kommunikation mit dem Automaten bzw. dem Hostsystem zuständig.**

**Code Completion -** Ist eine Technik, die dazu dient dem Programmierer Vorschläge zu machen, welche Funktionen oder Variablen er verwenden kann. Die IDE oder der Editor, der diese Funktion anbietet, analysiert was der Programmierer grade schreibt und macht seinerseits Vorschläge, zu dem was der Programmierer meinen Könnte, das spart Zeit durch das Vermeiden von Fehlern, die auf möglichen Schreibfehlern basieren.

**Compiler** - Ein Compiler ist ein Programm, das den vom Programmierer in einer bestimmten Sprache geschriebenen Quellcode in z.B. Maschinen Code umwandelt.

Das ist dazu nötig, damit die Zielmaschine das Programm versteht und ausführen kann.

Das Compiling erfolgt Zielsystem abhängig. Der Compiler muss also wissen, als welcher Art von Hardwarearchitektur das Programm später laufen soll, um so in einen passenden Maschinencode zu übersetzen.

**Debugger** - sind Werkzeuge für die Softwareentwicklung, die es ermöglichen Fehler in Programmen zur Laufzeit zu finden oder das Programm zur Laufzeit zu diagnostizieren.

**Embedded – zu Deutsch „eingebettet“ bezieht sich auf Verschiedene Bereiche.**

**In diesem Fall bezieht sich der Begriff auf das portieren eines Softwaresystems auf Embedded Computer, also möglichst schmale Rechensysteme, die gerade so viel leistung aufbringen um eine definierte Aufgabe abwickeln zu können.**

**Framework** - Als Frameworks sind Sammlungen von Funktionen also Bibliotheken zu verstehen, die dem Programmierer bereits festgelegte Funktionen zur Verfügung stellen.

Durch die Verwendung von Frameworks spart der Entwickler sehr viel Zeit, die er sonst damit verbringen müsste, Funktionen selbst zu schreiben, die häufig benötigt werden.

**Frontend – bezeichnet die „vordere“ Schicht eines Softwaresystems.**

**Das Frontend kann als Schnittstelle zwischen dem Benutzer und dem System angesehen werden. Das Frontend hat die Aufgabe dem Benutzer eine möglichst intuitive Oberfläche bereit zu stellen, die das bedienen eines Sysems vereinfacht und somit eine möglichst hohe akzeptanz erreicht.**

**GUI** - "Graphical User Interface"

Der Englische Begriff für grafische Oberflächen. Die dem Benutzer einer Software, einen möglichst effektiven Einsatz von Software ermöglichen soll.

**Hostsystem** – Das Hostsystem bezeichnet die Hardwareplattform also das Rechensystem auf dem die Software laufen soll.

**IDE** - Integrated Development Environment

Ist ein eine Sammlung von Programmen in einem Programmpaket oder Editor, der es dem Entwickler ermöglicht alle zum Entwicklungsprozess benötigten Werkzeuge in einer Oberfläche zu vereinen. Dies beinhaltet i.d.R einen Editor, Compiler, Debugger, Syntax Highlighting etc.

**Kiosk – bezeichnet die Verkaufsstelle bzw. im Kontext machner Software einen Modus der darauf ausgelegt ist auf speziellen Maschinen zu laufen. Im „Kioskmodus“ hat der Anwender nur noch eingeschränkte Möglichkeiten, kann z.B. das Programm nicht schließen oder andere Anweisungen ausführen. Das dient meist dazu eine nahtlose grafische oberfläche anzubieten.**

**MinGW** - Minimalist GNU for Windows ist die Portierung der GNU Compiler Collection auf Windows Systeme und ermöglicht dem Programmierer C, C++ und weitere Compiler unter Windows zu nutzen.

**POS – Point of Sale**

***Zu Deutsch Verkaufsort, bezeichnet aus Sicht des Käufers die Einkaufsstelle und aus Sicht des Verkäufers die Verkaufsstelle.***

**Proof of Concept - *Anhand eines Prototypen wird die prinzipielle Durchführbarkeit eines Vorhabens***

***getestet und je nach Ausgang eine Entscheidung über die weitere Durchführung eines Projektes gefällt.***

**Qt** - Ein sehr umfangreiches, starkes und gut dokumentiertes Framework für Programme mit vorwiegend grafischen Benutzer Schnittstellen. Wird derzeit noch von Nokia unterstützt, um Anwendungen der nächsten Generation auf Mobile Plattformen zu bringen. Darüber hinaus gibt es eine Vielzahl von Software, die mit Hilfe dieses Frameworks entwickelt wurden. Sowohl Wissenschaftliche als auch Militärische Einrichtungen verwenden Software auf dieser Basis. Dank dieses Frameworks kann heute selbst unter C++ noch effektiv und schnell programmiert werden.

**Semantic Navigation -** Diese Technik bieten manche moderne IDEs, um dem Entwickler Vorschläge zu machen, auch welche Methode oder Member variablen er in einem bestimmten Semantischen Kontext zugreifen kann. Diese Funktion zeigt den Entwickler schnell, ob er möglicherweise in einem beliebigen Umfeld noch keinen Zugriff auf bestimmte Methode oder Variablen hat. Dies spart zusätzlich Zeit, da der Entwickler nicht erst beim Kompilieren den Fehler entdeckt.

**Syntax Highlighting** - Ist eine Technik in modernen Editoren, die es ermöglicht Sprachspezifische Elemente Farblich hervor zu heben, um so dem Programmierer einen schnelleren Blick auf bestimmte Sprachelemente zu ermöglichen.

**Telemetrie – auch Fernmessung bezeichnet ein Verfahren, bei dem Daten an einem Ort aufgezeichnet werden und zu einer räumlich getrennten Sammelstelle übertragen werden. Die Daten können in der Sammelstelle entweder archiviert oder ausgewertet werden.**

**Vending – ist ein Begriff der im weiteren Sinne Verkaufsautomaten bzw. Selbstbedienungsautomaten bezeichnet.**

**Widget** - Als Widgets werden Komponenten grafischer Oberflächen bezeichnet, die den Benutzer die eigentlichen Schnittstellen zur Interaktion mit dem Programm dienen (Bsp. Eingabezeilen, Buttons, Textfelder etc.).

# 6 Quellangaben

# 7 Anlageverzeichnis