

TITEL

Seminararbeit

Version 0.01

Funktionale Programmierung
Zürcher Hochschule für Angewandte Wissenschaften

Simon Lang, Daniel Brun

Date

Versionshistorie

Version	Datum	Autor(en)	Änderungen
0.01	13.04.2015	DBRU	Initiale Version

Daniel Brun (DBRU)

Abstract

Ausgangslage und Ziel

Vorgehensweise

Detaillkonzept & Proof-of-Concept

Eigenständigkeitserklärung

Hiermit bestätige ich, dass vorliegende Semesterarbeit zum Thema „BigData mit RaspberryPi und F#“ gemäss freigegebener Aufgabenstellung ohne jede fremde Hilfe und unter Benutzung der angegebenen Quellen im Rahmen der gültigen Reglemente selbständig verfasst wurde.

Zürich, 24.09.2015

Simon Lang, Daniel Brun

Inhaltsverzeichnis

1	Einleitung	1
1.1	Hintergrund	1
1.2	Ziel	1
1.3	Aufgabenstellung	2
1.4	Erwartete Resultate	2
1.5	Abgrenzung	2
1.6	Motivation	2
1.7	Struktur	2
1.8	Planung	2
2	Recherche	3
2.1	Ausgangslage	3
2.2	Der Raspberry Pi	3
2.3	F# mit dem Raspberry Pi	4
2.3.1	Linux (Raspbian)	4
2.3.2	Windows 10 IoT	5
3	Datensammlung	6
3.1	Verwendete Hardware	6
3.1.1	GrovePi Sensoren	6
3.2	Verwendete Software	7
3.2.1	Datenauswertung	7
3.3	F# auf dem Raspberry Pi	7
3.3.1	Raspbian	7
3.3.2	Windows 10 IoT	7
4	Schlusswort	8
4.1	Fazit	8
4.2	Vergleich: Ist -/ Soll-Planung	8
4.3	Dank	8

Anhang	11
A Anhang	11

KAPITEL 1

Einleitung

Diese Arbeit wurde als Seminararbeit zur Vorlesung von funktionalen Programmiersprachen verfasst. In diesem Kapitel wird die Aufgabenstellungen und Rahmenbedingungen der Arbeit erläutert.

1.1 Hintergrund

Einer immer grösseren Beliebtheit erfreuen sich kleine Alltagsgegenstände welche mit dem Internet verbunden sind. Dieser Bereich wird IoT genannt. Diese Gegenstände sind in der Lage Daten zu erheben und weiterzuleiten. Da es zukünftig voraussichtlich immer mehr IoT Gegenstände geben wird fallen immer mehr Daten an. Diese Daten werden wegen ihrer Masse auch BigData genannt.

In dieser Arbeit wird evaluiert wie sich funktionale Programmiersprachen im Bezug auf IoT eignen, um BigData auszuwerten.

1.2 Ziel

Mit einem IoT Gerät sollen Daten aufgezeichnet werden. Diese werden als BigData gesammelt und sollen mit einer funktionalen Programmiersprache ausgewertet und ansprechend ausgegeben werden.

Das Hauptziel der Arbeit besteht darin zu überprüfen wie geeignet funktionale Programmiersprachen für die Auswertung von BigData sind. Als Nebenziel soll evaluiert werden, ob eine funktionale Programmiersprache zum erfassen von Daten auf einem IoT Gerät verwendet werden kann.

1.3 Aufgabenstellung

Die freigegebene Aufgabenstellung lautet wie folgt:

- Projektname: Seminar BigData mit RaspberryPi und F#
- Ausgangslage: Durch die rasante Entwicklung im Bereich IoT ergeben sich viele neue Anwendungsmöglichkeiten. Da der RaspberryPI immer leistungsfähiger geworden ist, soll evaluiert werden ob er sich für den Einsatz von funktionalen Sprachen im Bereich BigData eignet.
- Ziel der Arbeit: Es soll gezeigt werden wie F# Sharp auf einem RaspberryPi im Bereich BigData und IoT eingesetzt werden kann.
- Aufgabenstellung: Es soll gezeigt werden, wie eine funktionale Programmiersprache (F#) im Kontext von BigData und IoT eingesetzt und verwendet werden kann. Es soll eine Anwendung zur Sammlung von Sensordaten auf einem Raspberry PI und eine Anwendung zur Analyse / Auswertung der gesammelten Daten implementiert werden.

1.4 Erwartete Resultate

Gemäss freigegebener Aufgabenstellung werden folgende Resultate erwartet:

- Dokumentation
- Implementation / Prototyp

1.5 Abgrenzung

Aufgrund des Umfanges der Arbeit und der begrenzten Zeitdauer werden folgende Punkte von der Arbeit abgegrenzt:

- **Schnittstellendokumentation**
In dieser Arbeit werden nicht die Schnittstellendokumentationen und -spezifikationen rekonstruiert. Es werden jeweils die relevanten Aspekte betrachtet und hervorgehoben.

1.6 Motivation

1.7 Struktur

1.8 Planung

KAPITEL 2

Recherche

In diesem Kapitel werden die Grundlagen recherchiert wie die beiden Teilprojekte „Sensordaten sammeln“ und „Sensordaten auswerten (BigData)“ angegangen werden könnten.

2.1 Ausgangslage

Die Vorlesung zu diesem Seminar befasst sich mit den Konzepten der Funktionalen Programmierung. Zur Veranschaulichung dieser Konzepte wurde die Programmiersprache F# des .NET-Frameworks verwendet. Aufgrund dessen haben wir uns entschieden auch dieses Seminar mit der uns nun bekannten Sprache F# umzusetzen. Als IoT Gerät wird ein Raspberry Pi¹ verwendet. Der Raspberry Pi ist ein Einplatinencomputer welcher von der britischen Raspberry Pi Foundation entwickelt wurde. Der Raspberry Pi bietet den Vorteil, dass er sehr weit verbreitet ist², er kostengünstig ist und es inzwischen eine sehr grosse Anzahl an Sensoren auf dem Markt gibt mit welchem man Daten sammeln kann³.

2.2 Der Raspberry Pi

Wie bereits im vorangehenden Kapitel beschrieben, handelt es sich beim Raspberry Pi um einen Einplatinencomputer. Dieser Einplatinencomputer bietet verschiedene zentrale Hardware-Schnittstellen um externe Geräte für Input und Output anzuschliessen.

Vom Raspberry PI gibt es folgende Modelle:

- Raspberry Pi Compute Module
- Raspberry Pi Zero
- Raspberry Pi Model A

¹ [Raspberry_Pi_2016-04-24](#).

² [Raspberry_Pi_Erfolgsgeschichte_2016-04-24](#).

³ [Raspberry_Pi_Sensor_2016-04-24](#).

- Raspberry Pi Model A+
- Raspberry Pi Model B
- Raspberry Pi Model B+
- Raspberry Pi 2 Model B
- Raspberry Pi 3 Model B

Am 29 Februar 2016 ist die neuste Version, der Raspberry Pi 3 (Model B), auf dem Markt erschienen¹. Einige Zahlen zu dem Gerät:

- 1.2GHz 64-bit quad-core ARM Cortex-A53 CPU (10x die Leistung eines Raspberry Pi 1 und 50-60% die Leistung eines Raspberry Pi 2)
- Integriertes 802.11n wireless LAN und Bluetooth 4.1
- Komplette Kompatibilität zu Raspberry Pi 1 und 2 (Model B)

Evtl.
ein Bild
/ List
mit Da-
ten da-
zu?

2.3 F# mit dem Raspberry Pi

Um F# auf dem dem Raspberry PI auszuführen, gibt es grundsätzlich zwei Möglichkeiten, welche nachfolgend erläutert werden. Da es sich bei F# um eine Sprache des Microsoft .NET-Frameworks handelt, wird für die Ausführung zwingend eine Implementierung des .NET-Frameworks benötigt.

2.3.1 Linux (Raspbian)

Ein weit verbreitetes Betriebssystem für den Raspberry Pi ist das Raspbian² OS. Bei dem Namen handelt es sich um eine Zusammenfassung von Raspberry und Debian. Demnach handelt es sich auch um eine Debian Distribution, welche spezifisch für den Raspberry Pi entwickelt wurde.

Mono-Framework

Eine Möglichkeit um unter Linux, beziehungsweise Raspbian, F# auszuführen ist das Mono-Framework³. Dabei handelt es sich um eine Open Source Implementierung von Microsoft's .NET Framework.

¹ [Raspberry_Pi_3_2016-04-24.](#)

² [FrontPage_-_Raspbian_2016-04-24.](#)

³ [Mono_2016-04-24.](#)

Mono-Framework + .NET Core

Eine weitere Möglichkeit besteht aus einer Kombination des aktuellen Mono-Frameworks und der aktuellen Open Source Implementierung für Linux von .NET Core von Microsoft.

2.3.2 Windows 10 IoT

Microsoft hat mit Windows 10 IoT eine Version ihres Betriebssystems herausgebracht, welches speziell für leistungsschwächere Geräte entwickelt wurde¹. Bei der IoT Version von Windows 10 ist das .NET Framework bereits standardmäßig an Bord. Demnach sollte es keine Probleme geben um F# auf dieser Plattform zu betreiben.

Dokumentation
wir den
SW-
Setup
der
Hard-
ware?
Installationsan-
leitung?

¹ Windows_IoT_2016-04-24.

KAPITEL 3

Datensammlung

Dieses Kapitel beschäftigt sich mit dem Teilprojekt der Datensammlung. Dazu wird ein Raspberry Pi verwendet welcher mit F# Informationen von verschiedenen Sensoren zusammentragen soll. Diese Daten sollen in darauf folgenden Kapiteln weiterverwendet werden.

3.1 Verwendete Hardware

Für diese Seminararbeit wurden folgende Hardwarekomponenten verwendet:

- Raspberry Pi 2 Model B
- Raspberry Pi 3 Model B
- GrovePi Sensoren

3.1.1 GrovePi Sensoren

Für den Raspberry Pi gibt es viele Sensoren auf dem Markt. Herauskristallisiert hat sich jedoch das Starter Kit GrovePi+¹. Dieses beinhaltet unter anderem Ton-, Temperatur-, Feuchtigkeit- und Lichtsensoren. Der Vorteil an den GrovePi Sensoren besteht an den geringen Kosten, dem einfachen Anschluss an den Raspberry Pi und die vielen verfügbaren Beispiele in unterschiedlichsten Programmiersprachen.

Die Recherchen haben ebenfalls gezeigt, dass es eine Library für .NET gibt mit welchem die Sensoren angesprochen werden können².

¹ GrovePi_2016-04-24.

² NuGet__GrovePi_2016-04-24.

3.2 Verwendete Software

3.2.1 Datenauswertung

Die von den Sensoren gespeicherten Daten werden in einem noch zu definierenden Format abgespeichert und danach für die Datenauswertung ausgelesen. Dafür wurde vom Dozenten die Library F# Data vorgeschlagen¹. Diese kann Daten im Format CSV, HTML, JSON und XML entgegennehmen und für die Verwendung in F# zur Verfügung stellen.

3.3 F# auf dem Raspberry Pi

Im Abschnitt [2.3 F# mit dem Raspberry Pi](#) wurden zwei Möglichkeiten aufgezeigt, welche es ermöglichen F# auf einem Raspberry Pi laufen zu lassen. Entweder wird das Linux Raspbian mit Mono verwendet, oder das Windows 10 IoT.

Es war zu erwarten, dass der Weg über Window 10 IoT der einfachere ist, da dort .NET schon mitgeliefert wird. In diesem Projekt wurden beide Methoden ausprobiert. Die folgenden Abschnitte erläutern mit den beiden Herangehensweisen an das Problem.

3.3.1 Raspbian

3.3.2 Windows 10 IoT

Windows 10 IoT ist eine Version des Betriebssystems von Microsoft, welches speziell für kleinere Geräte mit weniger Rechenleistung konzipiert wurde.

Die Installation gemäss der Anleitung auf dem Github Account from Microsoft² war nicht erfolgreich. Der Raspberry Pi startete nicht und blieb beim Rainbow Screen³ hängen. Mit dem NOOBS⁴ Installer, welcher von der Raspberry Pi Foundation zur Verfügung gestellt wird, war die installation von Windows 10 Io

¹ Fsharp_Data_2016-04-24.

² install_win10iot_2016-04-25.

³ RPi_Rainbowscreen_2016-04-25.

⁴ NOOBS_2016-04-25.

KAPITEL 4

Schlusswort

4.1 Fazit

4.2 Vergleich: Ist -/ Soll-Planung

4.3 Dank

Abbildungsverzeichnis

ANHANG A

Anhang
