Die Entwicklung von Project Helix - Teil 0 - Vorstellung

by Drizzi - Donnerstag, Oktober 27, 2016

http://drizzisblog.net/2016/10/project-helix-teil-0/1356

Was ist Project Helix?

Project Helix (Gesprochen Hielicks) – Eine Software die das Leben einfacher gestalten soll. Sowohl online, als auch im realen Leben. Was genau ist Project Helix? Was ist der Sinn hinter der ganzen Geschichte und was hat Microsoft damit am Hut?

Project Helix ist mein kleines Schmuckstück. Ein Programm, welches wenig Fehler und kaum unsaubere Codezeilen enthält. Mittlerweile ist es bereits Vier Jahre alt und befindet sich seither in der Entwicklung.

Die Entstehung der Daten – Ein Darwin Wortspiel

Die Grundidee des Projektes Helix bestand darin ein Programm zu entwickeln, welche mir meinen Kaffee kocht und bringt. Ich schrieb mir ein wenig Code zusammen, bastelte an einer simulierten SPS die Pneumatik zusammen und steuerte eine kleine Eisenbahn an, welche mir dann meinen Kaffee bringen sollte. Alles sehr rudimentär, immerhin stand ich ja noch ganz am Anfang meiner Programmierkenntnisse. Ich wollte den Code bewusst sehr sauber halten und mich zu 100% an alle “Normen” halten, welche so in einer üblichen Programmierung sein sollten. CamelCase, Kommentare, Definitionen, etc. Summaries kannte ich damals noch nicht.
Ich lies das Programm dann Programm sein. Eine kleine Fingerübung unter vielen. Doch alles änderte sich als die Feuernation… ne Moment, falscher Film… Wo war ich? Ach ja…

Nach und nach experimentierte ich mit allen möglichen Klassen. Von FTP Servern über VPN Verbindungen bis hin zur simplen Sprachsteuerung und einem Möchtegern-Betriebssystem.

Es folgte, dass ich eines Tages in der Mittagspause feststellte, dass mein Heim-Netzwerk offline war. Das irritierte mich. Immerhin hatte ich den PC an gelassen. Zu dieser Zeit war es sehr warm und ich hatte tatsächlich Angst darum, dass mein Haus brennen könnte.

Vom Praktikanten zur Sekretärin – Project Helix wird befördert.

Irgendwann, in dem Jahr zwischen Erstentwicklung und meiner Paranoia auf Arbeit, hatte ich bereits den Gedanken gehegt Project Helix ein wenig mehr Tiefe zu geben. Die gewonnenen Erfahrungen mit einfließen zu lassen und ein sauberes großes Projekt zu schreiben, welches an Jarvis erinnert. Project Helix sollte also eine VI werden. Eine virtuelle Intelligenz. Ich benutze hier bewusst Mass Effect Terminologie, da ich die Unterscheidung zwischen einer intelligent wirkenden Software (Wie Siri, Cortana etc.) und einer wirklich “denkenden” Software wichtig finde. Zu dieser Zeit gab es Smart-Home

noch nicht wirklich kostengünstig und die Möglichkeiten waren relativ begrenzt. Die Vernetzung der Geräte im IoT waren damals, wie heute, recht bescheiden und mal ehrlich; Es kommt doch viel besser wenn man einfach mit seinem “Haus” oder seiner Technik reden kann, als dumm irgendwelche Handyknöpfe zu drücken.

Durch diese Zweifel an der Temperatur in meiner Wohnung hatte ich nun den Anstoß Helix wirklich existieren zu lassen. Ihr eine richtige Funktion zu geben. Ich schmiss den Kaffeecode raus und wandelte einige Klassen ab. Ich schrieb ein Interface zur Kommunikation und Steuerung von Phoner, sowie eine Klasse um alle relevanten Informationen aus dem Netzwerk auszulesen. Die Fritzbox half mir dabei Project Helix eine eigene Rufnummer zu geben und fertig war das Programm, welches ich benötigte. Es war gut. Project Helix ließ sich anrufen, stellte sich kurz vor und gab die Informationen preis die ich einprogrammiert hatte.

Uhrzeit, Netzwerkstatus Lokal, Netzwerkstatus Web und Kleinkram wie eben dieser. Project Helix war nun in der Lage meine Telefonanrufe zu beantwortet. Der Anrufer bekam zwar nur für ihn unwichtige Informationen, aber immerhin! Wer kann schon von sich behaupten eigene Sekretärin programmiert zu haben?^^

Scherz bei Seite, ich wollte mehr. Project Helix sollte Marktfähig werden. Sie sollte Interaktion simulieren und ein grobes Stück weit mit denken können. Doch zu dieser Zeit rechnete ich damit, dass ich Deep-Learning verwenden musste um eine wirklich brauchbare VI zu entwickeln und so verschob ich das Thema Interaktion auf einen späteren Zeitpunkt. Wenn Quantencomputer massentauglich wurden. Also begnügte ich mich vorerst damit Project Helix ein wenig über mein Umfeld bei zu bringen.

Ein Quantencomputer mit dem ich Helix das Deep Learning ermöglichen wollte. © Amherst College

Ich exportierte meine Android Kontakte und Telefonnummern in die Google Cloud, setzte eine Synchronisation auf und lies Project Helix die Kontakte einlesen und verarbeiten. Es war etwas Fummelei notwendig, jedoch konnte ich nun bereits in den Kontakten im Handy einstellen, ob der Anrufer, welcher über die Telefonnummer identifiziert wird, als Administrator registriert war und Einstellungen vornehmen, sowie weitergehende Informationen abrufen konnte nachdem er sich authentifiziert hatte, oder ob es nur ein Standartbenutzer war, welcher nur die aktuelle Uhrzeit und eine Information über meinen aktuellen Standort erhält. Ist der Benutzer überhaupt nicht erkannt worden, erhielt er nur eine kurze Zeitansage und bekam einen schönen Tag gewünscht.

Der heutige Stand

Project Helix ist auf Windows Basis in C# programmiert. Warum hat die Entwicklung nun so lange gedauert? Nun, das hat sie nicht. Nicht wirklich. Project Helix hat sich im Laufe der Jahre immer wieder geändert, so wie sich mein Programmierstil verändert hat. Es wurde Code neu geschrieben, neue Methoden verwendet, mit neuen Funktionen experimentiert und einiges auch wieder verworfen. In der Zwischenzeit wurde Project Helix auch mehrfach von Grund auf neu entwickelt, da ich beispielsweise in der ersten Version noch keine Klassen verwendet habe. Auch im Rahmen des nun anstehenden Projektes werden sicherlich viele Funktionen weg fallen und noch mehr neue hinzu kommen. aber darauf möchte ich im nächsten Teil eingehen.

Das Problem mit der Technik

Smart-Home Lösungen gibt es heutzutage leider wie Sand am Meer und der Traum einer der ersten zu sein die eine sehr bequeme Software und Hardwarelösung für die kleine Brieftasche entwickeln, dürfte wohl nicht mehr umsetzbar sein. Allerdings kostet ein Smart-Home Set immer noch einen Haufen Geld und viele Anbieter nutzen Abo-Modelle und schicken Daten dauerhaft an irgendwelche Web-Server. Das wollen viele nicht. Den großen Vorteil den ich, zumindest erst einmal nur für mich, in der Weiterentwicklung von Helix sehe ist, dass ich modular arbeiten kann und dennoch alle Geräte sinnvoll miteinander kombinieren kann. Klar, bei fertigen Lösungen wie FHEM oder Pilight kann man auch einiges einstellen und kommunizieren lassen, aber was bringt es mir, wenn die Jalousien herunter fahren wenn es zu warm draußen ist, die Heizung aber an springt? Die Sensoren müssen abhängig voneinander agieren können, sodass man nicht alles doppelt kaufen muss.

Ein kleiner Vergleich. Ein Smart-Home "Server" für die Steuerung des Lichts kostet bei Loxone 498€ (!). Und das war nur der erste Link dem ich gefolgt bin. Bei dem was ich mir vorstelle würde, nur die Hardware für die gleiche Funktionalität 12,49€ kosten… Dazu kommen noch die Empfangsgeräte, welche bei Loxone bereits 448€ kosten. Ein popliger Funkdimmer, den ich mit meinem System ansprechen kann kostet, (auf den ersten Blick) etwas bei 31€. Geräte wie einfache Kaffeemaschinen lassen sich ohne aufwändige Verkabelung bereits für (Im Schnitt) 6€ mit einem Empfangsgerät ausstatten.

Preislich sieht es bei Unterputzschaltern für das normale Licht ähnlich aus. Loxone bietet gar keine individuelle Lösung an, sondern benötigt erneut das Gerät für 448€. (Allerdings lassen sich damit sage und schreibe vier Lampen anschließen. Sofern diese nicht zu weit auseinander stehen. sonst braucht man einen Signalverstärker). Im Vergleich dazu sind 6-18€ für einen entsprechenden Schalter nun wirklich nicht zu viel verlangt…

Was nun ansteht

Ich werde Project Helix nun also Sehen, hören und fühlen bei bringen. Ebenso werde ich ihr einige Sinne mit geben, welche wir Menschen nicht haben. Einen Feuchtigkeitssensor, diverse Sender und Empfänger etc. Ich hatte als Basis einen Raspberry Pi (1) B angedacht, habe mir nun allerdings mit den Bauteilen die ich benötigte auch ein Arduino besorgt. Da ich auf einem Raspberry Pi, egal welcher Version, kein Windows Betriebssystem laufen lassen möchte, insbesondere dann nicht, wenn es mit dem Internet verbunden ist und mein Haus steuert, muss ich die Intelligenz von Project Helix, oder zumindest die Steuerung auf Java oder besser noch Python umschreiben. Sollte der Arduino sich als brauchbar für meine Zwecke erweisen, entfällt dieser Schritt, da die Portierung von C# in C++ deutlich einfacher möglich ist, als eine von C# in Java oder Python.

Der Hauptgrund weshalb ich den Arduino allerdings austesten möchte ist, dass er deutlich mehr Sendeleistung (5v <> 3,3V) besitzt als der Raspberry Pi. Auf diese Weise kann ich die Funkverbindungen weit genug aufbauen um keine weiteren Sensoren oder Repeater nutzen zu müssen. Und das bei einem kleinerem Energieverbrauch und effektiv mehr Leistung (Da kein Linux geladen werden muss). Der Preis spielt natürlich auch eine Rolle. Wir sprechen hier von einem 10€ Nachbau, im Vergleich zu einem

ähnlichen System welches bei 30€ beginnt. Gut, man könnte den Raspberry Zero einmal testen, aber da legt man auch gut und gerne 10€ hin und muss die Pins noch selbst anschrauben.

Auf lange Sicht ist natürlich eine Implementierung meiner Paranoia… äh, ich meine, der Krisenvorsorge vorgesehen. Ich denke an einen Rauchmelder, welcher selbstständig akustischen Alarm gibt, mir aber gleichzeitig eine SMS oder Push-Nachricht schickt. Das ganze lässt sich mit Sicherheit auch mit einem Seismographen und/oder Gasmelder koppeln.

Zunächst einmal möchte ich eine simple Funksteuerung für 433mhz Funksteckdosen realisieren. Basierend auf Programmgesteuerte Events oder eben Befehle. Sobald das alles funktioniert werde ich versuchen an günstige Richtmikrophone zu kommen, die allerdings auch brauchbar sind. Auf diese Weise kann ich meinem Haus dann Kontext und Zeitabhängig sagen “Ich bin müde”. Bis zu einer bestimmten Uhrzeit soll mir Project Helix dann einen Kaffee machen. Danach einmal kurz nachfragen was sie tun soll und bei entsprechender Antwort Rechner, Fernseher und Lichter löschen. Die Lichter natürlich nur dann, wenn die Bewegungssensoren gemeldet haben, dass ich nicht mehr im Raum bin

Project Helix ist ein Mammut Projekt. Es war als ein solches ausgelegt und wird auch als solches weiter betrieben. Sie wird nie perfekt sein, da ich immer wieder Funktionen, sinnvoll oder sinnlos hinzufügen werde. Sollte jedoch Interesse an dem Sourcecode bestehen, so werde ich diesen gerne für euch zur Verfügung stellen. Inklusive einer Einkaufsliste für die Hardware. Bedenkt allerdings, dass ich alle Sicherheitsrelevanten Dinge entfernen werde. Wäre ja dämlich, wenn ihr wüsstet wie ich meine Smartphone Zugänge generiere. Am Ende schaltet ihr mir noch das Licht aus wenn ich nicht hin schaue

Im nächsten Teil

Dieser Artikel diente hauptsächlich der Einführung von Project Helix. Eine kurze Vorstellung und ihre Hintergrundgeschichte. Im nächsten Teil gehe ich etwas konkreter auf die Vorstellungen ein die ich für das “fertige” Projekt habe. Davon wird vieles nicht in der mir zur Verfügung stehenden Zeit realisierbar sein, aber man kann ja trotzdem erst einmal Ideen sammeln. Des Weiteren werde ich, je nachdem welchen Umfang die Wunschliste annimmt, bereits auf die Hardware eingehen welche ich mir besorgt habe und wie sie zu verwenden ist.

Was hat Project Helix nun mit Microsoft zu tun? Ganz einfach. 2016 wurde ein Projekt zur Verbindung von Windows und X-Box vorgestellt. Namentlich “Project Helix”. Da ich kein Patent auf den Namen angemeldet habe, habe ich hier leider keine Befugnis oder rechtliche Gewalt. Da es zur Zeit auch keine eingetragene Marke ist oder kommerziell orientiert ist, wird mir Microsoft wegen des Namens sicher nicht an den Karren pinkeln können.

Project Helix ist aber, wie der Name bereits suggeriert, einfach nur die Projektbezeichnung gewesen. Meine VI, wenn sie es dann später einmal wird, hört auf den Namen “Helix” und ist somit außen vor. Ich muss an dieser Stelle nur beachten wie Viral dieses Project Helix von Microsoft wird, damit spätere Verwechslungen ausgeschlossen werden. Im schlimmsten Fall muss ich meine Helix, also mein Project Helix eben umbenennen in “Protocoll Helix” oder Ähnliches. Mal schauen. Wenn ihr mögt, teilt mir

Namensvorschläge gerne in den Kommentaren mit.

0110100001100101011011000110110001101111001000000111010001101000011001010111001001110010011100100111001001110010011100100111001001100101001000000111001101110100011100100110000101101110 0110011
1011001110110011101100111011001010111001000101110001000000100100100100000011000110110001101100011011000110110001101100011011000110110001101100101011001010110010100100000011110010110111101 1101

01001000000110110001101001011010110110010100100000011100100111001001110010011100100111001001110010011010010110010001100100011011000110010101110011001011100010000001111001011011110110111101110

1010111010100100000011000110110001101100011011000110110000101101110001000000110100001100101011011000110110001101100011011000110110001101100011011000110110001101100011100000010000001101101 0110

0101011001010010111000100000011010010010000001101000011000010111011001110110011101100111011001110110011001010010000001110100011011110010000001100101011110000110001101110101011101010111010 1011

1010101110101011101010111010101110011011001010010000001110100011010000110010100100000011011010110010101110011011100110111001101110011011100110111001101110011011000010110011101100101011100 1101

**1110101011001110110100000100000011100000110111101110111011101110111011101110111011101110111011101110111011101110111011101100101011100100010000001110100011011110010000001100110011011110111001001101101011000001
0111010000100000001110000001110010011011110111000001100101011100100110110001111001001011100010000000101010001101000011010010111001100100000001101101011011110111001001110010011100100111001001110011001
00111001001110010011011110110111000100000001101001011100110010000000111010001101111011011110010000000110001101101000011001010110000010111000000010000001110100011011110010000000110011101100101011101
000010000000110110101100101001000000011101000110100001100101001000000011010000110000010111001001110010011100100111001001110010011001000111011101100000101110010011001010010000000110100100100000001100
10001100101011100110111001101110011011100110111001101110011011100110110010101110010011101100110010100101110**

PDF generated by Kalin's PDF Creation Station