Lokaler Sprachassistent mit Dialogfähigkeit

Abb. 1: Aussen- und Innenansicht – Lokaler Sprachassistent

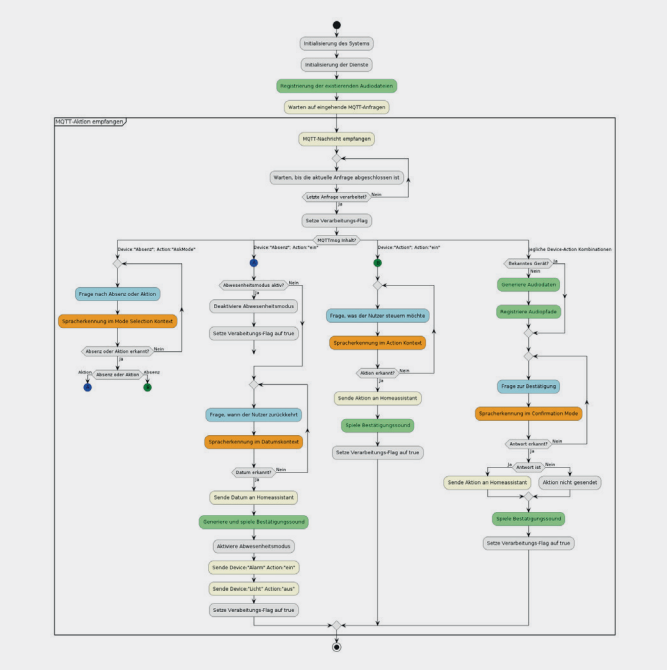

Abb. 2: Grobübersicht Programmablauf

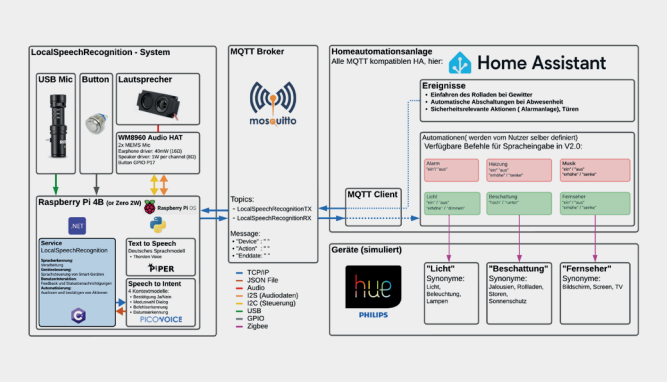

Abb. 3: Gesamtübersicht der Anwendung

Problemstellung

Typische Sprachassistenten benötigen eine permanente Internetverbindung, was in vielen Anwendungsfällen unerwünscht ist. Daher sind Sprachassistenten gefragt, die auch Datenverarbeitung in der Cloud funktionieren. In einer vorgängigen Arbeit wurde ein Assistent entwickelt, der «Ja» und «Nein» erkennen kann. Nutzer einer Homeautomation werden gefragt, ob eine Aktion ausgeführt werden soll, wie beispielsweise das Herunterlassen von Storen. Diese Anwendung bot bisher jedoch zu wenig Anwendungsmöglichkeiten und soll daher um eine Dialogfähigkeit zur Erfassung einer Abwesenheitsdauer erweitert werden, um im Homeautomationssystem gezielte Massnahmen ergreifen zu können.

Lösungskonzept

Das Konzept sieht vor, den Sprachassistent mit einem einfachen Dialogsystem zu erweitern, das auch komplexere Antworten verarbeitet. Dazu soll eine STI-Erkennung und TTS-Engine integriert werden, um eine flexiblere I/O zu erreichen. Durch die Modularisierung und ein Refactoring in spezialisierte Dienste, wird die Wartbarkeit und Erweiterbarkeit des Systems massgeblich verbessert.

Realisierung

Zu Beginn wurde die Testumgebung eingerichtet und die Softwarekomponenten installiert, einschliesslich der TTS und STI-Engine. Danach wurde die Modularisierung der bestehenden Klassen vorgenommen und neue Klassen zur Behandlung des Absenzdialogs implementiert. Eine flexible Architektur erlaubt es Geräte zur Laufzeit hinzuzufügen und ein robustes Fehlerbehandlungssystem implementiert mit NLog, erhöht die Stabilität und Zuverlässigkeit des Systems.

Ergebnisse

Die neue Implementierung führte zu einer signifikanten Verbesserung der Funktionalität. Der Sprachassistent kann nun flexibler auf Benutzeranfragen reagieren und auch Aktionen auslösen. Die Picovoice Rhino STI-Einheit ermöglicht eine präzisere und kontextbezogene Spracherkennung, während die Piper TTS-Engine dynamische Audiodateien zur Laufzeit generiert. Die modularisierte Architektur erleichtert die Wartung und Erweiterung des Systems erheblich. Diese Optimierungen führen zu einem verbesserten Benutzererlebnis.

Diplomand
Scheidegger Hannes

Dozent
Prof. A. Rumsch

Themengebiet
Technische Informatik (Embedded Systems)

Projektpartner
iHomeLab