# Aufbau einer Softwarearchitektur mit folgenden Spezifikationen:

### Step 1: Peripherieanalyse

1. **Such-Modul: Suche nach Geräten (IO-Link, FU, Alte Geräte)**
   * Such nach Hosts erfolgt über MAC-Adress-Anforderung und abgleich der MAC-Adresse mit Host-Typen in einer Datenbank
   * Benutzer wählt die zu durchsuchenden Hosts in der GUI aus ( **offen: brauchen wir einen Master Host ? )**
   * Realisierung der Suche nach IO-Link unter der Verwendung der Vorgehensweise von Javelin (Beckhoff spezifisch)
   * Alte Geräte werde über eine GUI eingegeben (evtl. Adressierung kann eingeteached werden)
   * Ansprechen der SPS über Treiberschnittstelle „SearchDevices“
   * Rekursives durchwandern der gefundenen Geräte nach etwaigen SubDevices
2. **Treiber-Modul: Laden der entsprechenden Treiber** 
   * Laden der zu den Devices zugehörigen Treiber aus einer Datenbank
   * Treiber haben einmalige Bezeichnung für jedes einzelne Gerät
3. **Kommunikations-Modul: Kommunikationsschnittstelle zum OPC UA Server** 
   * Definition der Treiber-Schnittstelle (Welche Daten fließen darüber?)
   * Integration der Treiberwerte in OPC UA Server
   * Bereitstellen der verfügbaren Skills
   * Definition der Datenschnittstelle (welche Elemente müssen über Skills kommuniziert werden, z.B. Geräte-ID und IP-Adresse bei NGD)
4. **VR-Modul: Virtual Representation**
   * Umwandlung des internen Peripherieabbilds in eine Struktur, die dem KommunikationsModul übergeben werden kann
   * Laden der Device-Skills
5. **Logging Modul: Loggen aller Programmmeldungen**
   * Statische Klasse, die von allen Programmteilen zugänglich ist und immer einen Eindeutigen Datenbestand aufweist
6. **GUI: Graphical User Interface**
   * Datenaustausch mit der BL erfolgt über das ViewModel
   * **Da das statische Databinding nicht funktioniert ist es eine Idee, das statische PropertyChanged-event im ViewModel abzufangen und per auswertung in entsprechende Propertys des ViewModels zu schreiben**

### Step 2: Konfiguration

1. **Kompilierungs-Modul Schreiben des PLC-ST auf die SPS**
   * Import des SFC (Structured-Flow-Chart)-Skill-Modells
   * Übersetzen der Befehle des allgemeinen ST-Codes in SPS spezifische Befehle
   * Verknüpfung von SFC und ST
   * Übergabe des Codes an SPS-Treiberschnittstelle „CompileCode“ -> anschließend konfiguriert der Treiber die einzelnen Tasks auf die SPS
   * Abgleich mit interner Konfiguration

### Spezifische Lösungen:

* Da der Broadcast-Search der Beckhoff-SPS bereits die meisten SubDevices findet, wäre es unsinnig für jedes weitere Layer die Suche redunat durchlaufen zu lassen
* USB-Geräte wie z.b. der Barcodescanner bekommen einen Treiber, der einen eigenen OPCUA-Server implementiert