# INTERACTION

Servo-Motor-1 -kommande

**Pi-camera**

**Or usb.web-cam**

**Video -frames**

**Android Application**

**Android**

**App**

**Java-Programm**

**Registration**

**Connection**

**HOST**

**Encrypter.cs (X)**

**Decrypter.cs (Y)**

*(commande -Connec*

**Raspberry pi box**

**Minibian-Os**

**Python / Java**

**Gps/Obd/Imu**

**DatenBank**

SENSORS RAW DATA Camera

*CSi-Connection*

App-User Connection

(Username + Password… Data)

*TCp-IP*

**Encrypter.php (Y)**

**Decrypter.php (X)**

*Tcp-Ip*

(Re)-Configuration

*GPIO*

(Servo-Motor-2 Kommande

*GPIO*

**Camera Images Provider Block**

*Tcp-Ip*

# Internal Software Architektur (Tcp-IP Client-Server Architektur ) :

**Übertragung nach dem Remote Server Server**

**Neue Konfigurationen**

**Java –Schnittstelle Interface zwischen dem Remote Server und**

**den lokalen Server (GPS-Provider,IMU-Provider, OBD-Provider)**

**TCP/IP Brücke**

**Socket Port: 4001**

**TCP/IP Brücke**

**Socket Port: 5001**

**TCP/IP Brücke**

**Socket Port: 3001**

**OBD - Server**

**Multi-Client OBD Provider**

**GPS - Server**

**Multi-Client Gps Provider**

**IMU - Server**

**Multi-Client Imu Provider**

**Bluetooth-Serial**

**Serial**

**I2C**

**Gps Sensor**

**Obd Sensor**

**Imu sensor**

# Prozess Ablauf-Diagramm + Boot-Sequenz

**Start**

**Boot-Sequenz Betrieb-system (MiniBian)**

**OBD-Server Starten :**

**Aktivierung der Bluetooth-Dongle RFcomm bind und Verbindung mit dem ELM327-Bluetooth Dongle oder Verbindung mit dem Obdsim Elm327 Simulator und Bereitstellung der Daten für die Client.**

**GPS-Server Starten :**

**Verbindung mit Arduino-Mikrokontroller durch Serielle Protokolle et Abfrage der GPPA-frames von GPS-module Sensor und Bereitstellung der Daten für die Client**

**IMU-Server Starten :**

**Verbindung mit dem GY-80 IMU Module durch i2C-Protokolle , Auslese der Werte von Gyroskope , Accelerometer, Barometer ,Temperatur und Druck-Sensor. Bereitstellung der Daten**

**Java-Interface Starten :**

* **Abonnement an der OBD , GPS und IMU Server und**
* **Dateien auf dem Remote-Server bereitstellen**
* **Neue Konfiguration von dem Remote-Server abrufen.**
* **Protokoll der letzten Dateiübertragung speichern**

# Internal Hardware Architektur

**Huawei e173 Modem**

**Wireless 3G Modem**

**HSDPA data transfer speed up to 7.2 Mpbs**

**UPS POWER SUPPLY**

**I2C**

**OBD-Simulator / Elm327 = OBD Schnittstelle**

RASPBERRY-Pi-A + Obdsim Software

+ Bluetooth Dongle

**IMU 9 Dof GY80**

[**HMC5883L**](http://www.sparkfun.com/products/10494) (triple-axis Magnetometer)

**L3G4200D** ( triple-axis Gyroskope )

**BMP085** ( Digital Pressure Sensor )

[**ADXL345**](https://www.sparkfun.com/products/9045) ( triple-axis accelerometer )

**Arduino Uno 8-bit Atmega Mikrokontroller**

**Serial**

**SPI**

## **GSM/GPRS/GPS** Shield Extension module

**USB/Serial**

**Zentrale Einheit**

**RASPBERRY-Pi-2**

**LINUX (ArchLinux / Minibian)**

**+**

**PYTHON + Java Code + Bash Script**

**Bluetooth-Serial**

[**ITG-3200**](http://www.sparkfun.com/products/9793) (MEMS triple-axis gyro)

**5 V/2A**