CanSat 2015 Team Gamma Dokumentation

Alexander Brennecke Till Schlechtweg Marc Huisinga Robin Bley
Steffen Wißmann Alexander Feldmann Kevin Neumeyer

19. Mai 2015

Inhaltsverzeichnis

1 Einleitung

1.1 Teamorganisation und Aufgabenverteilung

Das gesamte Team besteht aus sieben Schülern und zwei betreuenden Lehrern. Die sieben Schüler sind jedoch intern in mehrere kleinere Teams aufgeteilt. Innerhalb der Teams ist jedoch kein Teammitglied vollkommen an seine Aufgaben gebunden, da uns ein guter Austausch und eine hervorragende Zusammenarbeit zwischen den einzeln Teammitgliedern und Teams wichtig ist. Die Arbeit der Gruppen und der einzelnen Personen werden im folgenden erläutert:

- Das Hardware-Team besteht aus drei Personen, welche sich um den Bau des Satelliten selber, dem Design und dem Bau der Dose sowie der Programmierung des Mikrocontrollers kümmern. Zu diesem Team zählen folgende Personen:

  Alexander Brennecke ist verantwortlich für das Design der Dose. Dazu zählt die Konstruktion der eigentlichen Dose und die Anordnung der Sensoren im inneren der Dose.

  Till Schlechtweg ist verantwortlich für die Funktionalität des Mikrocontrollers und den ausgewählten Sensoren.

  Steffen Wißmann ist verantwortlich für die Übertragung der Daten zur Bodenstation und dem Programmcode des Mikrocontrollers.

- Das Software-Team besteht aus vier Personen, welche sich um das Programmieren der Bodenstation und der Android Applikation kümmern. Allgemein gilt für alle Personen dieser Gruppe, dass die Grenzen der Zuständigkeitsbereiche der verschiedenen Personen verfließen, wobei jede Person allerdings noch ein gewisses Spezialgebiet besitzt. Dieses Team besteht aus folgenden Personen:

  Robin Bley ist verantwortlich für das Implementieren der Datenverarbeitung der Bodenstation und für das Testen von kritischen Bereichen innerhalb der Datenverarbeitung.

  Alexander Feldmann ist verantwortlich für die Entwicklung der Android-Applikation.

  Marc Huisinga ist ebenfalls verantwortlich für das Implementieren der Datenverarbeitung der Bodenstation, für die Entwicklung der Datenvisualisierungskomponenten und für die Architektur der Datenverarbeitung.

  Kevin Neumeyer ist verantwortlich für die zusammenführende Architektur der Bodenstation, die grafische Umgebung und die Administration der Software-Repositories.

- Zudem gib es ein Team, bestehend aus Alexander Brennecke und Till Schlechtweg, zur Organisation, Kommunikation mit Sponsoren und Öffentlichkeitsarbeit.

- Betreut wird das Projekt durch zwei Lehrer unserer Schule:

  Mathematiklehrer Harm Hörnlein

  Physiklehrer Frank Marshall

Die Arbeit an dem Projekt findet zum größten Teil wöchentlich am Dienstag und Mittwoch Nachmittag in den Laboren unserer Schule statt. Die Labore sind mit diversen Werkzeugen ausgestattet, sodass sowohl die Software als auch die Hardware Gruppe dort problemlos arbeiten kann. Zusätzlich zu diesen vier bis acht Stunden pro Woche kommen fünf Projekttage, welche uns von der Schule gestellt wurden. Aber natürlich arbeitet jedes Teammitglied auch außerhalb dieser Treffen an seinem Fachgebiet, soweit dies möglich ist. Zusätzlich gibt es immer wieder Treffen mit externen Unterstützungen, oder Zeit in der Schule, wenn Vertretungs- oder Mitbetreuungsunterricht stattfindet.

1.1.1 Stärken des Teams

Die große Stärke des Teams ist es, dass es auch schon vor diesem Projekt existiert hat und sich somit sehr gut kennt. Die Teilnahme am europäischen CanSat Wettbewerb 2014 hat dazu geführt, das Ebenfalls von Vorteil ist, dass jedes Teammitglied durch unsere schulische Ausbildung genügend Wissen hat, um auch außerhalb seines Fachgebietes unterstützend tätig zu sein. Zudem ist die Arbeit- und Leistungsbereitschaft der meisten Teammitglieder überdurchschnittlich gut.

1.1.2 Verbesserungsbereiche des Teams

Der größte Verbesserungsbereiche des Teams liegt ganz klar im Zeitmanagement. Für viele Aufgaben wird zu wenig Zeit eingeplant. Oft kommt es auch vor, dass der Schwerpunkt der Arbeit auf Dingen liegt, die nicht höchste Priorität haben und somit Zeit beanspruchen, die eigentlich dringend woanders gebraucht wird. Ebenfalls problematisch ist, dass die meisten Teammitglieder gerne neue Technologien oder Praktiken ausprobieren wollen. Dieses Interesse ist zwar löblich und für die Einzelperson sehr lehrreich, jedoch kommt es bei neuen Technologien und Praktiken oft zu Problemen, die man bei bereits bekannten deutlich schneller lösen könnte.

1.2 Das Missionsziel

Die Idee hinter dem gesamten Projekts bezieht sich auf die extremen Umweltbelastung und ihre Folgen für den Menschlichen Körper. Ausschlaggebend für diese Idee ist ein Zeitungsartikel der Zeit, welcher über eine drohende Klage der EU-Kommission in Brüssel berichtet. (vgl. Die Zeit, 24.10.2014). Die EU-Kommission droht mit einer Klage gegen Deutschland, da die deutsche Bundesregierung bisher zu wenig Aufwand betreibt, um die Feinstaubkonzentration in der Luft zu reduzieren. Wir möchten diesen Aspekt aufgreifen und Messungen durchführen um die tatsächlichen Werte zu bestimmen. Der CanSat Wettbewerb eignet sich optimal dazu, da er uns die Möglichkeit bietet die Messungen nicht nur auf dem Boden sondern in verschiedenen Schichten der Atmosphäre durchzuführen. Feinstäube stehen in Verdacht, Krankheiten wie Asthma, Herz-Kreislauf Beschwerden und Krebs zu begünstigen.

Da der menschliche Körper nicht nur durch Feinstaub belastet wird haben wir uns entschlossen auch die Intensität der UV-Strahlung, welche die Hauptursache für Hautkrebserkrankungen ist, zu messen. Zusätzlich soll auch der Ozonwert bestimmt werden, da Ozon bereits in geringen Konzentrationen gesundheitsschädlich ist und zu Reizungen der Atemwege führen kann.

Für sich genommen ist jede dieser drei Größen schädlich für den Menschen. Im Zuge des Projektes wollen wir jedoch versuchen herauszufinden, ob es einen Zusammenhang zwischen ihnen gibt. Beispielsweise ist herauszufinden, ob ein höherer Ozon Gehalt gleichzeitig einen niedrigeren Feinstaubgehalt mit sich bringt.

Zusätzlich zum Bau des Messsystems im CanSat ist es unser Ziel eine einwandfreie Verarbeitung, Analyse und Präsentation der gemessenen Werte zu erzielen. Um dies zu garantieren programmieren wir ein eigenes Analysetool. Dieses Tool ermöglicht es uns die gemessenen Werte, während des Fluges des Satelliten, auszuwerten. Die Werte sollen dabei anschaulich und in Abhängigkeit zueinander dargestellt werden.

Um die Daten auch mobil verfügbar zu haben wollen wir eine Android Applikation bereitstellen. Diese Applikation soll vorerst nur für unser Projekt optimiert sein, bei Erfolg jedoch auch die Werte andere Teams anzeigen können.

1.3 Praktischer Nutzen für den Auftragsgeber

Die Ausrichter des Wettbewerbes, welche in unserem Projekt als Auftraggeber angesehen werden, können sich aus unserem Projekt wahrscheinlich relativ wenig praktischen Nutzen ziehen. Der Wettbewerb im Allgemeinen bietet den Veranstaltern jedoch mehrere Möglichkeiten. Zum einen können sie dadurch mehr Jugendliche für die Bereiche Raumfahrt, Elektrotechnik, Informatik und Physik begeistern. Zum anderen kann es auch möglich sein, dass Mitglieder der Jury, welche

meist in einem der gerade genannten Fachbereiche tätig sind, Lösungsansätze für Probleme der Wissenschaft in einem der Projekte wiederfinden.

2 Beschreibung des CANSAT

2.1 Missionsüberblick

Wir haben uns für den Satelliten überlegt, dass dieser so weit wie möglich individuell sein sollte. Daher greifen wir nicht auf das, vom Wettbewerb bereitgestelte T-Minus CanSat Kit zurück. Stattdessen haben wir uns im Detail überlegt, welche Sensoren unseren Erwartungen entsprechen und wie wir diese bestmöglich innerhalb der Dose plazieren können. Zusätzlich möchten wir nicht auf eine Cola-Dose als Hülle zurück greifen, sondern möchten auch hier unser eigenes Design erschaffen.

2.2 Mechanisches und Strukturdesign

Wir haben den CanSat in drei Komponenten aufgeteilt: Die Hülle, die Innenwand und die Sensorik Platine. Diese drei Komponenten bilden den Hauptbestandteil des CanSats und haben maßgeblich zu dem mechanischen und strukturellem Design beigetragen. Im nachfolgenden wird kurz auf jeden dieser Komponenten eingegangen und die exakte Funktion im Zusammenhang erklärt.

2.2.1 Die Hülle

Wir haben uns dazu entschieden, die äußere Hülle aus GFK (Glasfaser verstärkter Kunststoff) zu fertigen. Dieses hat die Eigenschaften, dass er bei einem sehr geringen Gewicht, und bei einer geringen Wandstärke trotzdem eine gewisse Stabilität aufweißt. Aus dem GFK haben wir eine Röhre mit einem Innendurchmesser von 31,5 mm und einem Außendurchmesser von 33,5 mm laminiert. Diese Röhre wurde auf eine Länge von 111 mm gekürzt und gefeilt. Um die Röhre oben und unten zu verschließen haben wir uns bei Thyssen Krupp System Engineering zwei Aluminium Deckel fräsen lassen. Diese haben uns ebenfalls durch ihr geringes Gewicht und ihre hohe Stabilität überzeugt.

2.2.2 Innenwand

Um die Elektronik innerhalb der Hülle zu platzieren und zu befestigen haben wir uns dazu entschieden eine Wand anzufertigen. Diese Wand teilt die Hülle mittig und bietet so auf beiden Seiten Platz um unser Mikrocontroller Board und unsere Sensorik Platine zu befestigen. Beide Bauteile werden mittels vier Gewindestangen an der Wand befestigt. Durch die Technik des 3D-Druckens ist es möglich der Wand ein sehr geringes Gewicht bei einer verhältnismäßig hohen Stabilität zu verleihen. Zusätzlich gibt es uns die Möglichkeit die Wand millimetergenau zu gestalten.
Am unteren Ende der Wand befindet sich eine Aushöhlung, sowie ein Fuß. Diese ist zum einen dafür da um den Sharp Feinstaub Sensor zu befestigen. Zum anderen gibt der Fuß der Wand und somit dem gesamten Satelliten eine gewisse Stabilität. Der Fuß besitzt auf der einen Seite der Wand Bohrungen. Diese Bohrungen werden verwendet um die Aluminiumdeckel an der Wand zu befestigen. An der oberen Seite der Wand befinden sich ebenfalls solche Bohrungen um den oberen Deckel der Hülle zu befestigen. Da der Feinstaubsensor einen Luftzug benötigt befindet sich ein Durchlass innerhalb der Wand. Um das Mikrocontroller Board mit der Sensorik Platine zu verbinden existiert ein Fenster in der Mitte der Wand. Um die Sensorik Platine und das Mikrocontroller Board an der Wand zu befestigen existieren vier Bohrungen.

2.2.3 Die Sensorik Platine

Die Sensorik Platine ist eine von uns geätzte Platine, welche mit unseren Sensoren bestückt ist. Es gibt mehrere positive Aspekte, die eine eigene Platine mit sich bringt. Zum einen bietet sie eine stabile Plattform für die Befestigung der Sensoren. Zum anderen sparen wir uns dadurch eine Menge Kabel, welche deutlich störanfälliger sind als eine Platine. Die Platine hat an den entsprechenden Stellen Bohrungen um sie mit der Zwischenwand und dem Mikrokontrollboard zu verbinden. Die Platine bietet Platz für folgende Module:

- BMP108 Drucksensor: Misst den Luftdruck und gibt diesen, sowie die daraus berechnete Höhe zurück
- Sparkfun UV Sensor: Misst die Intensität des Spektrums 270-380 nm, welches dem UVA und UVB Spektrum entspricht
- TMP006 Infrarot Temperatursensor: Misst die Temperatur eines dünnen Aluminiumstückes in der Außenwand
- Adafruit Ultimate GPS: Bestimmt die aktuelle Position sowie die Höhe
- APC220 Transceiver Modul: Sendet die Daten als JSON String zur Bodenstation
- Steckplatz zum Anschluss des Sharp Feinstaub Sensors: Misst den Anteil der Partikel, welche kleiner als 10 µm sind
- Steckplatz zum Anschluss an das Mikrokontrollerdboard: Bildet die Schnittstelle zwischen BeagleBone und Senorik Platine

2.2.4 Fachliche Grundlage

Um die 3D gedruckte Wand zu erzeugen wurde die 3D Moddelierungssoftware Sketchup von Google verwendet. Sketchup bietet die Möglichkeit vergleichsweiße einfach 3D Modelle zu zeichnen. Um dies zu tun muss klar sein, welche Objekte gezeichnet werden sollen. Diese Objekte müssen vermessen und innerhalb von Sketchup gezeichnet werden. Dies erfordert die Kentniss über gewisse mathematische Methoden zur Berechnung von Kreisen, Flächen und Körpern. Die meisten 3D-Drucker benötigen Datein des Types .stl, welche in Sketchup mit einem Plugin erzeugt werden können. Zum fertigen von GFK Komponenten wird ein Körper benötigt, auf welchen das GFK laminiert werden kann. In unserem Fall ist dieser Körper zylindrisch, mit einem Durchmesser von 31,5 mm, und aus Aluminium gefräst.
Um die Platine zu erstellen wurde die Design Software Eagle PCB verwendet. Eagle bietet die Möglichkeit sowohl Schaltpläne als auch das entsprechende Layout zu erstellen. Im Anschluss wurde die Platine, mit Hilfe und Mitteln des Hackerspace Bremen e.V. geätzt.

2.3 Elektrische Konstruktion

2.3.1 Fachliche Grundlagen

2.3.1.1 Embedded System Ein Embedded System ist in unserem Fall der BeagleBone mithilfe vom ARM Cortex-A6 mit 1Ghz, ist ein leich modifizierte linux kernel mit Frontend installiert, dass liebevoll Angstrom genannt wurde. Um ein paar andere Beispiele für ein eingebettetes System sind etwa ein Smart TV oder ein Router, beide haben eine Art eingebettetes System, dass immer öfter auf dem Linux Kernel basiert und je nach Anwendung angepasst wurde. In unserem Fall unterstützt das BeagleBone verschiedene Technologien zum empfangen von Daten verschiedener Bauteile, wie etwa UART, I-2-C, SPI, Analog, Digital, PWM, Timer und PRU. Viele dieser Technologien sind in unserem Projekt nicht in Verwendung alle anderen Grundlagen sind unten beschrieben.

2.3.1.2 Transistor-Transistor-Logik 5V werden immer als logische 1 bezeichnet, damit ist gemeint wenn der Sensor den höchsten Messwert erreicht gibt er eine Spannung von 5V. Ist dies nicht der Fall hat der Sensor eine andere Kennkurve die zum Beispiel bei 3.3V aufhört. Allgemein wird aber Transistor-Transistor-Logik genutzt, welche 5V als logische 1 und geerdet als logische 0 ansieht, es gibt natürlich Toleranzen, diese sind aber bei verschiedenen integrierte Schaltkreisen und Mikrokontrollern unterschiedlich.

2.3.1.3 Analog-to-Digital-Converter Andere Sensoren wie der UV-Sensor die nur über einen internen Wiederstand verfügen der sich, je nach Konzentration, an einer mathematischen Kurve orientierend, im Wert leicht verändert und dadurch die ankommende Spannung am jeweiligen Analog Pin ändert. Mithilfe eines Analog-to-Digital-Converter konvertieren wir das analoge Signal, zum Beispiel 5V, in das äquivalente digitale Signal mit der Auflösung von 12 Bits.

$$2^{12} = 4096$$

$$\frac{5V}{4096} = 0.001220703125V$$

Das bedeutet jeder 0.001220703125V kann dargestellt werden, wobei der Arduino Mega 2560 nur 10 Bits zur Verfügung stellt.

$$2^{10} = 1024$$

$$\frac{5V}{1024} = 0.0048828125V$$

Der Arduino kann den Wert der am analogen Pin ankommt viel ungenauer darstellen, als das BeagleBone.

2.3.1.4 Universal-Asynchronous-Receiver-Transmitter UART ist eine digitale serielle Schnittstelle zum realisieren von einfachen Kommunikationen zwischen zwei Endpunkten, die Funktionsweise ist denkbar einfach. Wir nutzen in unserem Satelliten meist eine Baudrate von 9600bps, Baud ist die Schrittgeschwindigkeit oder Symbolrate, also 9600 bits per second. Für UART gibt es wie beim RJ45 Stecker TX und RX, die beim Aufbau einer Kommunikation gekreuzt werden. Transciever und Reciever. Nun wird zwischen vielen verschiedenen Arten von UART unterschieden in unserem Fall die TTL-UART Variante welche die beim Analog-to-Digital-Converter genannten 5V als logische 1 bezeichnen.

2.3.1.5 Inter-Integrated-Circuit I-2-C ist ein serieller Datenbus der über zwei Kabel mit einer 10-Bit-Adressierung, 1024 IC's steuern kann mit einer maximalen Geschwindigkeit 5 Mbit/s. Der Sinn des Bussystems ist es mithilfe von einer Adressen einen Datensatz oder Befehl nur an den gewünschten Empfänger zu senden, obwohl nur eine Datenleitung genutzt wird, eine Art Master/Slave System. Der Master sagt wer wann zu sprechen hat und welche Befehle von wem zu empfangen sind.

2.3.1.6 Python Als Programmiersprache zur Programmierung des Beaglebone Black's, haben wir uns für die Programmiersprache Python entschieden. Es wäre zwar ebenfalls möglich gewesen den Mikrocontroller mit den Sprachen JavaScript, Java, C, C++, C# und vielen weiteren Sprachen zu programmieren, da es sich bei dem Beaglebone um ein embedded-System handelt, welches praktisch alle Programmiersprachen unterstützt, sofern entsprechende Librarys existieren. Allerdings haben wir uns aufgrund der Tatsache, dass Python im Gegensatz zu Java nicht objektorientiert geschrieben werden muss, und wir auf der Hardwareseite möglichst auf objektorientierte Programmierung verzichten wollen, da sie nicht nötig ist, für Python entschieden. Ein weiteres wichtiges Argument war die gute Python-Library, welche von einer großen Community permanent gewartet und aktualisiert wird.

2.4 Softwareentwurf

2.5 Bergungssystem

3 Beschreibung der Bodenstation

3.1 Überblick

In diesem Teil der Dokumentation werden wir die Bodenstation vorstellen, welche als Datenempfänger und als Datenverarbeitungsplattform fungiert.
Die Bodenstation wurde von Robin Bley, Marc Huisinga und Kevin Neumeyer entwickelt.

Die zentrale Aufgabe der Bodenstation ist es, die Daten, welche vom Satelliten gesammelt werden, zusätzlich sicher am Boden zu speichern, sollte der Satellit und damit auch die lokal gespeicherten Daten verloren gehen.
Zusätzlich zur Datensicherung erfüllt die Bodenstation die Aufgabe, die empfangenen Daten auf verschiedene Arten zu visualisieren und somit dem Nutzer direkt während der Datenübertragung die Möglichkeit zu verschaffen, die Daten zu beobachten und diese zu analysieren.
Die Bodenstation ermöglicht es außerdem, dass gesicherte Daten auch nach der Datenübertragung noch betrachtet und analysiert werden können.
Unser Ziel bei der Entwicklung der Bodenstation war es, eine modulare und anpassbare Plattform zu entwickeln, welche nicht nur mit unserem Satelliten, sondern mit vielen verschiedenen Satelliten genutzt werden kann, ohne dass ein großer Konfigurationsaufwand besteht.
Um dies zu ermöglichen, haben wir die Bodenstation in mehrere Dimensionen skalierbar entwickelt, was es im Endeffekt sehr einfach macht, neue Satelliten und verschiedene Übertragungsprotokolle zur Bodenstation hinzuzufügen.

3.2 Verwendete Komponenten

Zum Erreichen unserer Ziele haben wir verschiedene Komponenten verwendet, welche einerseits der Datenvisualisierung und -analyse dienen, andererseits aber auch der Entkopplung und skalierbaren für die Entwicklung dienen.

Für die Bodenstation haben wir folgende Komponenten verwendet:

Java ist eine objektorientierte Programmiersprache. Diese wurde verwendet, da jedes unserer Gruppenmitglieder damit vertraut ist. Die Version Java 8 wurde verwendet um mächtige funktionale Features zu nutzen

Netbeans Platform das die Möglichkeit bietet, einfach eine integrierte, modulare und entkoppelte GUI-Applikation auf Basis von Java Swing zu entwickeln

JUnit ist ein Framework, welches zum Erstellen von automatisierten Softwaretests dient.

JSerialComm (zum Start des Projektes noch serial-comm) ist eine Bibliothek, welche das Auslesen serieller Schnittstellen ermöglicht

NASA World Wind ist eine Software, welche Satelliten- und Luftbilder auf einem virtuellen Erdball darstellt. Daten der Bodenstation werden mittels dieser Software in Relation zur Höhe in Echtzeit visualisiert.

JChart2D ist eine Grafik-Bibliothek, welche zur grafischen Visualisierung von Daten dient. Mithilfe dieser Bibliothek werden zweidimensionale Graphen erzeugt, welche empfange Daten des Satelliten, in Relation zur Zeit oder anderen Daten, in einem Graphen darstellt

JSON (JavaScript Object Notation) ist ein Datenformat, welches zum austausch von Daten zwischen Anwendungen angewand wird. JSON ermöglicht es Daten in verschiedenen Format in Textform zu speichern und sie wieder zurück in ihre uhrsprüngliche Form zu interpretieren. Dieses Datenformat wird in der Bodenstationsoftware genutzt Daten mit dem Sateliten auszutauschen, zu loggen, zu exportieren und zu importieren.

3.3 Funktionen

3.3.1 Nutzerfreundlichkeit

Die Bodenstation wurde so entwickelt, dass der Nutzer der Bodenstation sich nicht um Implementationsdetails scheren muss und die Bodenstation als zentralen Empfänger für Daten von seinem Satelliten nutzen kann, ohne dabei etwas anderes als die grafische Benutzeroberfläche zu verwenden.

Ein wichtiger Faktor im Bereich der Nutzerfreundlichkeit ist die dynamische Benutzeroberfläche, welche es dem Nutzer erlaubt, verschiedene GUI-Komponenten in Teilpanels innerhalb der Applikation anzuzeigen und umzustellen. Diese Dynamik ermöglicht es dem Nutzer, die Benutzeroberfläche, welche für die Analyse seiner Daten am Besten ist, mithilfe der verschiedenen Panels einzustellen.

Ein weiterer, wichtiger Faktor ist, dass der Nutzer die Bodenstation so anpassen kann, wie es für die Datenübertragung seines Satelliten am Besten ist. Alle Teilkomponenten der Datenübertragung sind über die grafische Benutzeroberfläche austauschbar, was es sehr einfach macht, die Bodenstation auf die Datenübertragung des jeweiligen Satelliten anzupassen.

Alle Visualisierungskomponenten sind zudem intuitiv aufgebaut, sodass es nicht kompliziert ist, sich seine Daten mithilfe der Visualisierungskomponenten anzusehen. Die Anzeige über den 3D-Globus erlaubt es beispielsweise, den Globus beliebig zu bewegen und die Position auf dem Globus zu verändern, während der Graph es erlaubt, dass die Axen des Graphen beliebig ausgetauscht werden können.

3.3.2 Erweiterbarkeit

Bei der Entwicklung der Bodenstation haben wir darauf geachtet, dass die Bodenstation auf einer skalierbaren Architektur aufgebaut ist. Dies ermöglicht es, leicht neue Module und Funktionalitäten zur Bodenstation hinzuzufügen, ohne dabei besonders viel Code abzuändern. Auf die folgenden Weisen ist die Bodenstation skalierbar:

- Neue GUI-Komponenten können sehr leicht hinzugefügt werden. Das Erstellen und Einbinden eines GUI-Komponenten umfasst lediglich die Erstellung eines neuen Netbeans-TopComponents.
- Unterschiedliche Satelliten können ohne eine erneute Kompilierung hinzugefügt und verändert werden, indem man die Konfigurationsdateien der Bodenstation anpasst, welche zur Laufzeit von der Bodenstation geladen werden.
- Neue Konfigurationsformate können leicht hinzugefügt werden, indem man die neue Konfigurationsimplementation unter einem Interface in der Applikation hinzufügt.
- Es ist ohne viel Aufwand möglich, die verschiedenen Datenquellen, aus denen Daten bezogen werden, auszutauschen. Möchte man also die Daten von einer anderen Quelle als einem USB-Port beziehen, so ist dies leicht zu implementieren, indem man ledigliche eine neue DataSource implementiert und zur Applikation hinzufügt.
- Verschiedene Datenübertragungsformate können ebenfalls ohne viel Aufwand hinzugefügt werden, indem neue Formate unter dem DataFormat-Interface implementiert werden. Die Applikation ist also nicht auf JSON als Übertragungsformat limitiert.
- Die verschiedenen Datenempfänger können auch leicht angepasst werden, indem man neue Datenempfänger unter dem Receiver-Interface implementiert, wodurch die verschiedenen Logging-Formate erweitert werden können.
- Daten können aus beliebigen Dateiformaten importiert werden, was ebenfalls leicht erweiterbar ist, indem man neue Import-Dateiformate über das Importer-Interface implementiert.
- Export-Formate können leicht erweitert werden, indem man neue Export-Formate unter dem Exporter-Interface implementiert.

3.3.3 Features

Da die Software der Bodenstation auf dem Framework Netbeans Plantform basiert, lassen sich einzelne graphische Module kombinieren, welche sich per drag and drop verschieben lassen. Die Größe und Position dieser Module und des gesamten Frames lassen sich beliebig verändern. Des weiteren bietet die Software die Möglichkeit, Daten von verschiedenen Satelliten zu empfangen. Empfangene Daten lassen sich mittels der graphischen Oberfläche in Graphen anzeigen, welche sich verschieden kombinieren lassen. Außerdem lassen sich die empfangenen Daten zwischenspeichern und anschließend in verschiedene Dateiformate exportieren oder live in einer Datei loggen. Diese exportierten oder geloggten Daten lassen sich anschließend wieder einlesen und anzeigen. CSV, TXT, JSON, KML und PNG sind Dateiformate, welche exportierbar sind. Davon lassen sich exportierte CSV- und JSON Dateien wieder einlesen und visualisieren. TXT-Dateien werden formatiert und somit gut leserlich für den Nutzer exportiert, während exportierte KML-Dateien per Google Earth geöffnet und graphisch visualisiert werden können. Ein weiteres Feature der Bodenstationsoftware ist die Datenvisualisierung per Nasa World Wind als Modul in der graphischen Oberfläche. Diese Visualisierung zeigt den Flug des Satelliten auf einem virtuellen Globus, mittels Satelliten- und Luftbilder, und zeigt auf jeder gemessenen GPS-Koordinate die gemessenen Werte der Sensoren des Satelliten. Unter anderem bietet dieses Modul der Software die Möglichkeit, an die virtuelle Erdkugel heranzuzoomen und einzelne Elemente dreidimensional darzustellen. Darstellungen mittels dieses virtuellen Erdballs sind sowohl in Echtzeit mittels eines Streams vom Satelliten als auch als Import aus einer Datei möglich.

3.4 Architektur

Insgesamt ist die Architektur der Applikation um das GUI-Framework Netbeans Platform aufgebaut, da es die Nutzung von bestimmten Architekturen einfacher macht, mit Netbeans Platform zu arbeiten und alle Features von Netbeans Platform zu nutzen.

Insgesamt ist die Applikation in Module und Pakete aufgeteilt. Während normale Java-Projekte normalerweise lediglich in Pakete aufgeteilt sind, werden in Netbeans Platform die einzelnen Komponenten normalerweise in Module aufgeteilt. Jedes Modul verhält sich hierbei wie ein einzelnes Projekt, welches dann vom Hauptprojekt eingebunden wird. Dies fördert generell die Wiedernutzbarkeit der einzelnen Module, da sich Entwickler darüber Gedanken machen müssen, wie die einzelnen Module in verschiedenen Umgebungen verwendet werden können.
Den Aufbau der Netbeans Platform Modularchitektur ist auch im Anhang unter Abbildung 3 zu finden.

Anfänglich haben wir jeden einzelnen Teilkomponenten in ein Modul ausgelagert, was jedoch zu einer Menge Merge-Konflikten geführt hat, da Netbeans Platform für jedes einzelne Modul eigene Konfigurationsdateien generiert, über welche man nur schwer den Überblick behalten kann. Diese anfänglich Architektur wurde schlussendlich in eine Architektur mit nur drei Modulen umgewandelt: API, Core und GUI.
Das Core-Modul enthält die Programmlogik, welche sich hauptsächlich mit der Verarbeitung der Daten innerhalb der Input-Pipeline beschäftigt.
Das GUI-Modul enthält die verschiedenen GUI-Komponenten, welche sowohl Teil der allgemeinen Benutzeroberfläche sind, als auch Visualisierungskomponenten darstellen.
Innerhalb des API-Moduls befinden sich Interfaces und Utility-Klassen, welche sowohl vom Core-Modul als auch vom GUI-Modul genutzt werden. Das Core-Modul implementiert hierbei die Interfaces aus dem API-Modul, während das GUI-Modul die Programmlogik im Core-Modul lediglich entkoppelt über die Interfaces des API-Moduls anspricht.
Diese Architektur zeigt bereits, dass das GUI-Modul vom Core-Modul entkoppelt ist. Diese Entkopplung trägt stark zu der Erweiterbarkeit der Applikation bei. Die Architektur ähnelt hierbei der standardmäßigen "Model, View, Controller"-Architektur, jedoch scheint innerhalb der Architektur kein Controller vorhanden zu sein. Auf den ersten Blick gesehen scheint es so, als spreche das GUI-Modul das Core-Modul direkt über das API-Modul an, jedoch sorgt Netbeans-Platform dafür, dass dem nicht so ist. Die von Netbeans Platform bereitgestellten Lookups, welche eine

weitere Ebene der Entkoppelung darstellen, erfüllen in der Applikation die Aufgabe des Controllers. Durch die Lookups wird innerhalb des GUI-Moduls eine passende Klasse innerhalb des Core-Moduls über das Interface der Klasse im API-Modul geladen. Dank den Lookups und den Interfaces im API-Modul besteht absolut keinerlei Kopplung zwischen den einzelnen Klassen und Modulen: Die GUI kennt das Model nicht und das Model kennt die GUI nicht.
Die Modularchitektur der Bodenstation ist ebenfalls im Anhang unter Abbildung 4 zu finden.

3.5 Tests

3.5.1 Automatisierte Tests

Einzelne Komponenten der Software werden mittels automatisierten Tests per JUnit getestet. Dabei werden bei den jeweiligen Export-Komponenten jeweils Testdaten in das jeweilige Format exportiert. während dessen kann überprüft werden ob sich die Komponente, bei der Übergabe verschiedener Parameter, wie geplant verhält. Außerdem kann geprüft werden, ob die erzeugten oder veränderten Dateien wie geplant aussehen. Darüber hinaus wurde für jeden Import-Komponenten ein automatisierten Test geschrieben, welche den Komponenten auf das verhalten bei verschiedenen Parametern überprüft. Des weiteren werden Daten erzeugt, welche zunächst mit dem jeweiligen Komponenten exportiert werden und anschließend mit dem passenden Komponenten importiert werden. Dabei wird geprüft, ob sich die importierten Daten von den ursprünglichen Daten unterscheiden.

3.5.2 Manuelle Tests

3.6 Nutzeranleitung

3.6.1 Livestream starten(Daten von einem Satelliten empfangen)

3.6.2 Daten importieren

3.6.3 Graphen anpassen

3.6.4 Daten exportieren

3.6.5 Daten mittels WLAN an Clients weiterleiten

3.6.6 Graphische Oberfläche personalisieren

3.7 Kosten-/Nutzenanalyse

3.8 Realisierung

Während der Realisierung der Bodenstation kam es zu einigen Komplikationen. Zum einen wurde die Softwarearchitektur während der Implementationsphase geändert, so dass verschiedene Komponenten wie zum Beispiel Export- und Importkomponenten mehrfach realisiert wurden. Diese Architekturveränderung wurde vorgenommen und gewisse Komplikationen zu beseitigen, welche die Modulare Strukturierung von Netbeans Plattform mit sich bringt. Wir starteten mit einer Architektur, welche jede wichtige Komponente als Modul benennt. Diese Architektur brachte zum einen das Problem, dass Abhängigkeiten zwischen Modulen nur in eine Richtung stattfinden kann. Die neue Architektur unterscheidet lediglich zwischen den Modulen API, GUI und Core. Des weiteren wurden verwendete Datentypen innerhalb der Software während der Implementationsphase geändert, so dass zusätzlich alle Komponenten, welche mit der Datenverarbeitung Zutun haben geändert werden mussten. Darunter vielen die gesamten Komponenten des Imports, Exports und der Live-Datenverarbeitung. Die Veränderung der benutzten Datentypen wurde von Long, String und Double zu einzig Double geändert, da alle Daten, welche von kompatiblem Satelliten in Double dargestellt werden können. Diese Änderung im Programmcode hat zwar einiges an Arbeit gekostet, doch der Umfang des Programmcodes wurde deutlich verringert und die Performance gesteigert.

4 Projektplanung

4.1 Zeitplan der CanSat Vorbereitung

Die Zeitplanung ist ausgerichtet für den Zeitpunkt der Abgabe unseres P5, da wir uns gewünscht haben, zu diesem Zeitpunkt mit dem Projekt fertig zu sein. Dieser Zeitplan wurde jedoch von Anfang an sehr kritisch gesehen. Daher ist es nicht verwunderlich, dass der Fortschritt des Projektes geringer ist, als er zum jetzigen Zeitpunkt eigentlich seien sollte. Dies ist jedoch nicht dramatisch, da bis zum Wettbewerb genügend Zeit ist die restlichen Arbeitspakete abzuarbeiten. Das gesamte Management der Arbeitspakete und des Zeitaufwandes wurde mit der Projektmanagementsoftware Redmine erledigt. Da diese auf unserem Server unter redmine.gamma-team.de erreichbar ist kann jedes Teammitglied zu jedem Zeitpunkt den Fortschritt der Arbeit verfolgen. Die Planung der beiden Halbgruppen ist größtenteils voneinander getrennt. Es gibt jedoch gemeinsame Meilensteine, welche von beiden Gruppen eingehalten werden sollen. Bevor die Arbeit der Halbgruppen begonnen hat gab es eine allgemeine Projektfindungsphase. In dieser Phase wurde ein grober Zeitplan festgelegt und es wurden alle relevanten Systeme (Webserver, Projektmanagementsoftware, GitLab etc.) aufgesetzt und eingerichtet um später einen reibungslosen Ablauf der Arbeitsphase zu garantieren. Die Idee und die Spezialisierung der Idee für das gesamte Projekt entstand ebenfalls in dieser Zeit. Anschließen wurde eine separate Zeitplanung in den beiden Halbgruppen erstellt, welche im Nachfolgenden erläutert wird.

4.1.1 Zeitplan der Hardware Gruppe

Innerhalb der Hardwaregruppe wurden versucht die meisten Aufgaben zu parallelisieren. Jedes Teammitglied hat sein eigenes spezielles Aufgabengebiet. jedoch herrscht trotzdem ein stetiger Austausch zwischen den Teammitgliedern. Grund für die Parallelisierung war, dass in unseren Augen die meisten Aufgaben nur die Aufmerksamkeit einer Person benötigen. Es ist nur selten erforderlich, dass mehrere Teammitglieder an ein und dem selben Arbeitspaket arbeiten. Der gesamte Arbeitsprozess wurde in diverse Abschnitte gegliedert. Diese Abschnitte lassen sich auch im GANTT Diagramm im Anhang dieses Dokumentes wiederfinden. Bei den Abschnitten handelt es sich um folgende:

- Planung: Erstellung von Arbeitspaketen, sowie eine Verteilung dieser und eine Erstellung diverser Diagramme
- Fallschirm: Gestaltung und Bau des Bergungssystems.
- Sensorik: Dieser Abschnitt behandelt das Heraussuchen, Bestellen und Testen passender Sensoren für unser Projekt.
- Beagleboard: Festlegung der Programmiersprache, IDE und der Recherche zu den elektrotechnischen Eigenschaften des Boards
- Dose: Design und Bau der Hülle und der Deckel der Dose
- Dosenmanagement: Design und Bau des inneren der Dose, sowie die Integration der Sensoren in das Gesamtsystem

Die einzelnen Abschnitte sind in diverse Arbeitspakete unterteilt, Personen zugewiesen und mit einem Zeitraum versehen.

4.2 Einschätzung der Mittel

4.2.1 Budget

Um das CanSat Projekt zu finanzieren konnten wir aktuell noch keine Sponsoren finden. Jedoch konnten wir uns mit unserem Schulverein verständigen, welcher uns finanziell unterstützen wird.

Da wir nicht auf das T-Minus Kitt zurückgreifen sondern stattdessen ein anderes Mikrocontroller Board verwenden können wir ungefähr 150 €sparen. Der 200 €Watterot Gutschein, welcher vom Wettbewerb gestellt wird, ist in unseren Rechnung noch nicht inbegriffen. Dies liegt daran, dass noch nichts bei Watterot bestellt wurde, bzw. die Bestellung lange vor der Annahme am Wettbewerb getätigt wurde. Im Nachfolgenden sind alle Ausgaben und Einnahmen aufgelistet.

Tabelle 1: Ausgaben

Ausgabe	**Datum**	**Empfänger**	**Grund**
-12,16 €	08.01.2015	Watterott	BMP180 Breakout
-28,99 €	09.01.2015	eBay - rcskymodel	Ultimate GPS
-14,32 €	10.01.2015	Spark Fun Electronics	UV-Sensor
-51,99 €	10.01.2015	Amazon	Beagle Bone Black
-17,30 €	01.12.2014	eBay - hdt-preiswert	GFK-Set 1kg Polyesterharz + 20g Härter + $2m^2$ Glasfasermatte
-3,54 €	23.03.2015	toom baumarkt	6 x Schleifpapier
-3,79 €	23.03.2015	toom baumarkt	Filzrolle
-4,49 €	23.03.2015	toom baumarkt	Plüschwalzen
-2,19 €	23.03.2015	toom baumarkt	Mundschutz
-1,99 €	23.03.2015	toom baumarkt	Farbwanne
-4,99 €	23.03.2015	toom baumarkt	Einmalhandschuhe
- 145,75 €			

Tabelle 2: Einnahmen

Einnahmen	**Datum**	**Absender**	**Grund**
17,30 €	01.12.2014	Alexander Brennecke	GFK-Kauf
107,46 €	10.01.2015	Alexander Brennecke	Sensorenkauf
20,99 €	23.03.2015	Alexander Brennecke	toom Einkauf
145,75 €			

4.2.2 Externe Unterstützung

Externe Unterstützung erhielten wir von vielen Lehrern unserer Schule, welche uns Fragen zur Elektrotechnik und Softwareprogrmmierung beantworten konnten. Zusätzlich haben wir finanzielle Unterstützung durch den Schulverein unserer Schule erhalten (siehe 4.2.1). Unterstützung außerhalb unserer Schule erhileten wir durch folgende Personen/Organisationen:

- Das Hackerspace Bremen e.V., welches uns ihren 3D-Drucker zur Verfügung gestellt hat. Zusätzlich konnten wir dort unsere Platine ätzen.
- Prof. Martin Schneider von von dem Hochfrequenzlabor der Universität Bremen, welcher uns geholfen hat unsere Antenne an die Frequenz und die Wellenimpedanz anzupassen.
- Das Umweltlabor der Atlas Elektronik GmbH hat uns geholfen den CanSat, hinsichtlich seiner Stabilität, zu testen und die Sensoren korrekt zu kalibrieren.

4.2.3 Testkonzept

5 Öffentlichkeitsarbeit

5.1 Website

Unsere Website Team Gamma wurde bereits für den europäischen CanSat Wettbewerb 2015 verwendet. Diese haben wir weiter geführt und dort in unregelmäßigen Abständen aktuelle Informationen über das Projekt veröffentlicht. Da die Website durch den europäischen Wettbewerb bei anderen europäischen Teams bekannt ist wird die Website in Englisch geführt. Man findet dort zusätzlich einige Dokumente, Fotos und Videos. Die Informationen auf der Website sind meist relativ detailliert verfasst.

5.2 Schülerzeitung

In der Schülerzeitung unserer Schule sind bereits diverse Artikel über unser Projekt erschienen und sollen auch in Zukunft erscheinen. Diese Artikel handeln zumeist von dem Wettbewerb selber und gehen weniger auf die technischen Details ein.

5.3 Präsentationen

Da wir das CanSat Projekt bereits seit einiger betreiben kommt es immer wieder vor, dass wir es vor unserer Klasse präsentieren. Dies kommt zum Beispiel dann vor, wenn wir Teile des Projektes in Schulprojekte einfließen lassen. Zusätzlich haben wir, beispielsweise am Tag der offenen Tür unserer Schule, diversen Schulbesuchern das Projekt und den Wettbewerb näher gebracht.

5.4 Ausstellung am MINT-Projekttag unserer Schule

Im Schuljahr 2015/2016 findet an unserer Schule ein Tag der MINT Projekte statt. Dieser Tag wird von einer Schülergruppe unserer Parallelklasse organisiert und wir wollen an diesem Tag natürlich unser Projekt vorstellen.

5.5 Logo

Das Logo wurde ebenfalls aus Gründen der Wiedererkennbarkeit aus dem vorherigen Jahr übernommen. Das Aussehen des Logos wurde von drei Faktoren beeinflusst:

- Das Zeichen in der Mitte soll dem Gamma Logo ähneln, welches zu unserem Teamnamen passt
- Das Zeichen soll zusätzlich, wenn man es um 180° dreht, dem Lambda Logo ähneln. Da bei dem Entwurf unserer Antenne immer wieder auf Lambda gestoßen sind, sind daraus diverse interne Späße entstanden, die wir in das Logo einfließen lassen wollen.
- Das Logo des Computerspiel Halflife, welches von einigen Teammitgliedern gespielt wird

6 Anforderungen

7 Reflexion des Projektverlaufes

7.1 Reflexion der Hardwaregruppe

Als wir angefangen haben das gesamte Projekt zu planen haben wir uns als Ziel gesetzt Ende Mai fertig zu sein. Dieses Datum haben wir Aufgrund der Abgabe unseres P5 gewählt, für welches wir das CanSat Projekt ebenfalls einreichen wollen. Uns war bewusst, dass dies ein sehr hoch gestecktes Ziel ist. Im Nachhinein haben wir relativ schnell gemerkt, dass wir dieses Ziel nicht erreichen können. Diese Verzögerung wurde durch mehrere Faktoren hervorgerufen. Dazu zählt der enorm hohe Anspruch den wir uns selber gesetzt haben. Dieser hatte immer wieder zur Folge, dass viele Dinge mehrfach oder gründlicher gemacht werden mussten, als es zu Anfang geplant war. Zum anderen haben wir verhältnismäßig lange gebraucht um uns auf eine finale Idee festzulegen und diese zu präzisieren. Da wir uns jedoch kontinuierlich zum arbeiten getroffen haben konnten wir dennoch gute Fortschritte erzielen. Wir lagen zwar die meiste Zeit über hinter unserem Zeitplan, konnten jedoch die Reihenfolge der zu bearbeitenden Aufgabenpakete größtenteils einhalten.

7.2 Reflexion der Softwaregruppe

7.3 Reflexion der Zusammenarbeit zwischen den Teams

Da der inhaltliche Schwerpunkt der beiden Teams relativ wenig miteinander zu tun hat sollte es theoretisch relativ wenige Berührungspunkte geben. Dies war bei unserer Projektarbeit jedoch nicht so. Da die Arbeit der beiden Halbgruppen zur gleichen Zeit in der gleichen Räumlichkeit stattfand war es oft so, dass teamübergreifend diskutiert wurde. Dies hat den Vorteil, dass beide Teams nochmal einen anderen Blick auf eventuelle Problemstellungen bekommen und so einfache oder bessere Lösungen für Probleme finden können. Zusätzlich lief die Absprache über den Datenaustausch zwischen Bodenstation und CanSat sehr gut. Die beiden Teams haben also hervorragend kooperiert und gemeinsam versucht ein bestmögliches Gesamtprodukt zu erschaffen.

8 Anhang

8.1 GANTT-Diagramme

8.1.1 Hardware-GANTT

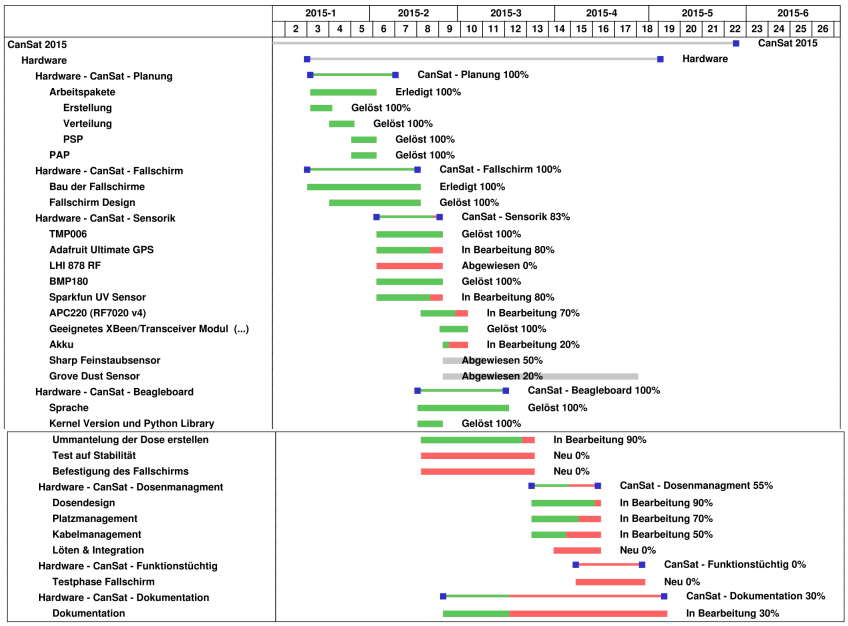

Abbildung 1: Das GANTT-Diagramm der Hardware Gruppe

8.2 Der CanSat

Abbildung 2: Der Satellit (Diese Zeichnung ist möglicherweiße nicht sichtbar, da es eine 3D Zeichnung ist. Bitte verwenden sie den Adobe Acrobat Reader

8.3 Bodenstationsarchitektur

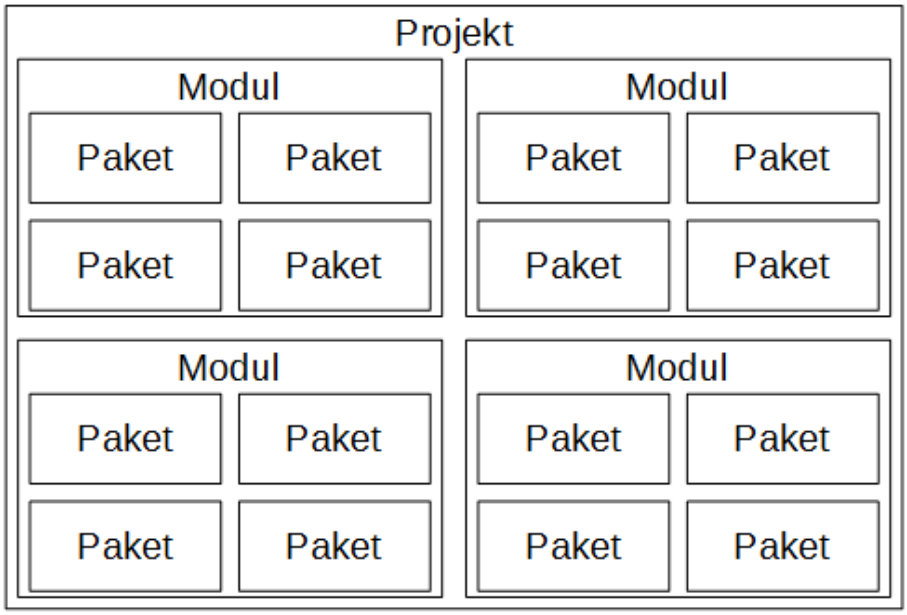

Abbildung 3: Modularchitektur von Netbeans Platform

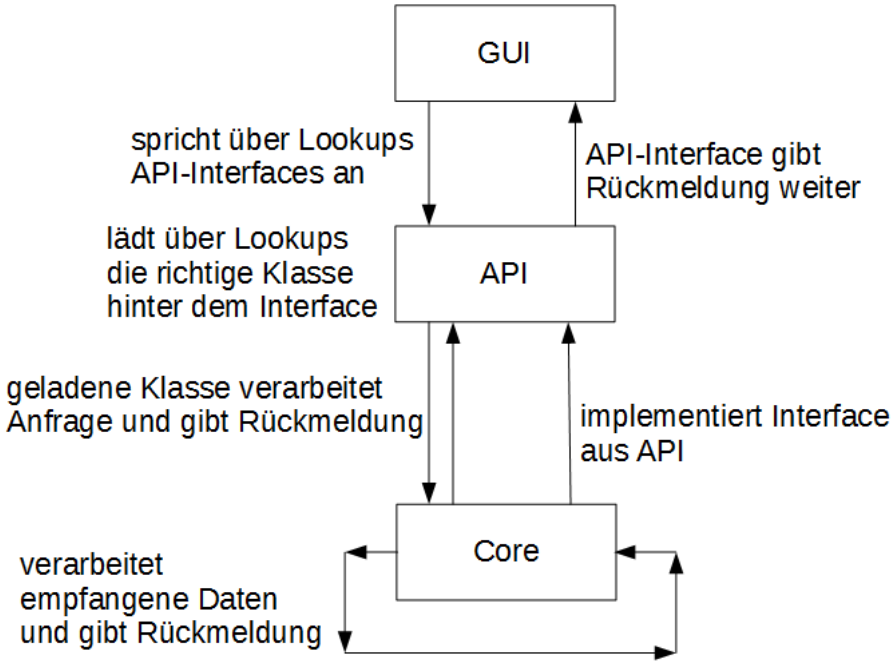

Abbildung 4: Modularchitektur der Bodenstation