0.1 Produktübersicht

Der **FR**aunhofer**O**penSource**S**ensor**T**ings-Server des Fraunhofer IOSB ist eine Open-Source-Implementierung des SensorThings-Standards des OGC und unter https://github.com/FraunhoferIOSB/FROST-Server downloadbar. Der FROST-Server dient der Speicherung und Verwaltung von Sensordaten.
Unsere Software stellt zu diesem Server eine einfache und dennoch stark konfigurierbare Weboberfläche bereit, die es ermöglicht Sensordaten aus einer Tabelle einzulesen, zu konvertieren und auf einen FROST-Server zu übertragen. Die Software läuft in einem Java JRE und das zugehörige Webinterface soll über einen herkömmlichen PC, einen Laptop oder ein Tablet erreichbar sein.

0.2 Musskriterien

1. Import von Tabellen
   Die Daten sollen aus Tabellen im CSV-/XLSX-Format importiert werden

2. Konvertierung des Formats der Sensordaten
   Die Daten sollen aus der Tabelle in eine einheitliches Format nach Vorgabe der SensorThings API konvertiert werden. Damit entsprechen sie dem Format der Daten auf einem FROST-Server.

3. Erstellen, abspeichern, laden von Konfigurationen
   Die Konfiguration der Software (ausgew. Thing, Server, Format) soll gespeichert und dem Nutzer via Download einer CFG-Datei bereitgestellt werden. Eine Konfiguration soll später auch wieder geladen werden können.

4. Datentyptransformationen
   Ein Wert als bspw. String soll nicht als String auf den FROST-Server übertragen werden sollen, sondern von der Software vorher in einen passenden Datentyp (bspw. Integer) umgewandelt werden

5. Verarbeiten von Sonderwerten (Magic Numbers)
   Diese Werte sollen von der Software ignoriert und gegen einen NULL-Wert (o.ä.) ausgetauscht werden, bevor sie auf den FROST-Server übertragen werden.

6. Import von Fremdquellen wie anderen Webseiten
   Es soll möglich sein als Quelldatei auch einen entfernten Speicherort -zum Beispiel einen Weblink- anzugeben.

7. Weitergabe von Fehlermeldungen an Nutzer, falls Import schief läuft
   Bei Fehlern in der Verarbeitung oder in der Übertragung soll der Nutzer eine Rückmeldung kriegen

8. Logging
   Sämtliche Eingaben und Interaktionen des Nutzers mit der Software sollen in einer Log-Datei gespeichert werden

0.3 Wunschkriterien

1. Sprachauswahl: Deutsch/Englisch

2. Bereitstellen einer Vorauswahl
   Dem Benutzer wird eine vorgefertigte Standard-Konfiguration bereitgestellt.

3. Überprüfung auf Duplikate
   Beim Upload der Daten auf den FROST-Server werden diese mit den bestehenden Daten abgeglichen und Duplikate werden nicht erneut hochgeladen.

4. Korrektur von falschen Daten
   Der Benutzer hat dei Möglichkeit Fehler in den Daten direkt korrigieren.

5. Rückgabe nicht bearbeiteter Datensätze in neuer CSV-Datei
   Bei (teilweise) fehlgeschlagener Übertragung zum FROST-Server werden die nicht übertragenen Daten in einer CSV-Datei an der Benutzer zurückgegeben.

6. Docker-Container
   Zur einfachen Installation steht das Programm als Docker-Container zur Verfügung

7. Auswahl komplexer Transformationen Aggregationen (Zusammenlegen von Daten) Aggregationen wie Summe, Minimum/Maximum, Durchschnitt, ... über ausgewählte Daten werden dem Benutzer in einem Webinterface bereitgestellt.

8. automatisierte Daten-Aktualisierung
   Bei Import von Daten von einer entfernten Adresse wird eine automatische regelmäßige Übertragung bereitgestellt.

9. Import kompletter Datensätze von anderem FROST-Server
   Der Benutzer hat zusätzlich die Möglichkeit, Daten von einem anderen FROST-Server

10. automatisierte Erkennung des Formats
    Aus der CSV-/XLSX-Datei werden Formate automatisch erkannt.

0.4 Abgrenzungskriterien

1. Keine Verwaltung von Sensor-Daten auf dem FROST-Server
   Die Software soll keine schon auf dem FROST-Server liegenden Daten verwalten, ändern oder löschen

0.5 Anwendungsbereiche

Überall wo Daten im CSV-Format verarbeitet werden müssen:
Sie Software soll überall dort eingesetzt werden wo Daten aus Tabellen importiert, konvertiert und auf den FROST-Server übertragen werden sollen.
Hierzu zählt:

- Der Import von neuen Sensordaten, welche in CSV-/XSLX-Tabellen-Form vorliegen, in den FROST-Server

- Die Integration von bestehenden (archivierten) Sensordaten, welche in CSV-/XLSX-Tabellen-Form vorliegen, in den FROST-Server

0.6 Zielgruppen

Die Software ist auf zwei Zielgruppen ausgelegt.
Zum einen gibt es die Personen, welche einfach Sensordaten auf einen Frostserver laden möchten, ohne die Konfigurationen zu ändern. Der Fokus liegt

bei dieser Gruppe auf der einfachen Bedienung der Software.
Zum anderen gibt es die Nutzergruppe welche ihre eigenen Konfigurationen erstellen. Für diese Nutzergruppe ist es wichtig, dass sie komplexe Einstellungsmöglichkeiten in den Konfigurationen nutzen können.

0.7 Betriebsbedingungen

Unterschieden wird zwischen Endgerät und Server.
Die Betriebsbedingungen beschreiben die minimalen Anforderungen die die Software zur Ausführung benötigt.
Das Ziel-Endgerät ist ein Computer/Tablet, wobei jedes Gerät mit einer Internetanbindung zum Programmserver, einem Webbrowser und aktiviertem JavaScript ausreichend ist.

Der Programmserver benötigt nur eine Anbindung an den FROST-Server und das Endgerät und ein Java JRE.

- Browser
- JavaScript
- Verbindung zum Programmserver
- Verbindung zum FROST-Server
- Java JRE
- Anbindung an Endgerät (zB Internet)

0.8 Produktumgebung

Die Produktumgebung ist eine Instanz unter der die Software läuft. Diese erfüllt mindestens die Betriebsbedingungen.

- head-less Server
- Linux/Windows Server
- Java JRE
- SensorThings API

Die Software soll plattformunabhängig auf einem Server mit minimaler Rechen- und Speicherkapazität laufen. Des weiteren soll das Programm head-less ununterbrochen laufen. Der Server stellt das Java JRE zur Verfügung, auf dem die Software läuft. Die Software benötigt nur noch eine Verbindung zu einer FROST-Server-Instanz, welche ihm vom Nutzer in der Konfiguration gegeben wird.

0.9 Funktionale Anforderungen

- Kurzanleitung auf der Weboberfläche: Auf der Weboberfläche existiert die Möglichkeit, eine Anleitung angezeigt zu bekommen. Diese soll kurz beschreiben, wie der Konfigurator und die Weboberfläche an sich funktionieren.

- Auswahlmöglichkeit des Zielservers: Auf der Weboberfläche soll ein erreichbarer Frost-Server zum Hinterlegen der Daten einstellbar sein

- Auswahlmöglichkeit der Datenquelle: Auf der Weboberfläche lässt sich auswählen, ob eine Datei (.csv/.xlsx) hochgeladen, eine entfernte Web-Adresse als Quelle dient oder Daten aus einem schon bestehenden Server importiert werden sollen.

- Auswahlmöglichkeit einer Konfiguration: Auf der Weboberfläche besteht die Möglichkeit eine Konfigation abzuspeichern und eine abgespeicherte Konfiguration erneut zu laden.

- Bereitstellen einer Default-Konfiguration

- Konfiguration zum Laden der Daten auf den Frost-Server: Auf der Weboberfläche lassen sich das Format der hochzuladenden Daten einstellen. Dazu wird die Auswahl bzw. das Neuanlegen eines Things und zugehörigen Datastreams ermöglicht. Außerdem lässt sich das Datumsformat in der Daten in der Quelldatei festlegen.

- Hochladen der Daten auf den Frost-Server

- Formatieren der Daten: Vor dem Abspeichern auf dem Frost-Server werden die Daten formatiert, sodass sie dem OGC SensorThings API Standard enstprechen.

- Anlegen einer Log-Datei: Während der Nutzung der Weboberfläche wird eine Log-Datei erstellt, die sämtliche Aktionen des Nutzers sowie Weboberfäche/Server-Interaktionen aufzeichnet, um Fehlererkennung und -verfolgung zu erleichtern.
- Sprachauswahl
- Auswahlmöglichkeit komplexer Operationen auf Daten: Auf der Weboberfläche lassen sich komplexere Datentransformationen auswählen. Diese sollen das Abspeichern von Datenaggregationen wie Summe, MIn, Max und Durchschnitt des Datensatzes ermöglichen
- Duplikaterkennung: Bevor Daten auf den Frost-Server hochgeladen werden, wird überprüft, ob diese schon existieren, um redundante Daten zu vermeiden.
- Fehlermeldungen: Kommt es bei der Verarbeitung der Daten zu Problemen, ermöglichen aussagekräftige Fehlermeldungen eine effiziente Korrektur.
- Rückgabe nicht verarbeiteter Daten: Können Teile des Datensatzes nicht verarbeitet werden, wird eine csv-Datei mit allen fehlerhaften Daten erstellt und an den Nutzer zurückgegeben.

- Hilfe-Button: Durch Klick wird die README-Datei im Browser aufgerufen
- Speichern-Button: Bei Klick auf diesen Button wird der Upload der Daten in die Datastreams/MultiDatastreams auf den FROST-Server angestoßen
- Radio-Boxen "Datei auswählen"/"Web-Adresse angeben": Durch Auswahl einer der Boxen wird entweder die Textbox zur Eingabe des Weblinks der entfernten Datei freigeschaltet oder der "DurchsuchenButton zur Auswahl einer lokalen
- Durchsuchen: Durch Klick auf den "DurchsuchenButton öffnet sich der Datei-auswählen-Systemdialog, bei dem man die Datei auswählen kann. Im Systemdialog wird das Dateiformat auf entweder CSV oder XLS/XLSX begrenzt. Nachdem die Datei im Systemdialog ausgewählt wurde, wird sie auf die Größe überprüft und bei erfolgreicher Prüfung hochgeladen.
- Textbox SZiel": Hier wird die Adresse des FROST-Servers angegeben
- Button Äuswählen": Durch Klick auf diesen Button wird die im Textfeld SZielängegebene Adresse überprüft und durch einen Haken und im Log bestätigt. (Es sollte überprüft werden ob eine Verbindung hergestellt werden kann,ob eventuell Zugriffsrechte benötigt werden und ob die Zieladresse überhaupt zu einem FROST-Server führt)
- Konfiguration durchsuchen/laden: Durch Klick auf "Durchsuchen"öffnet sich der Datei-auswählen-Systemdialog in dem man die CFG-Konfigurationsdatei auswählen kann. Die Konfiguration wird geprüft und anschließend geladen.
- Konfiguration speichern: Durch Klick auf den Button SSpeichern"wird der Download der aktuellen Konfiguration als CFG-Datei gestartet
- Reset-Button: Bei Klick auf den Reset-Button wird die gesamte Konfiguration (Datums-/Zeitformat, Datastreams, Ziel) zurückgesetzt
- Thing- / id-Dropdown-Menü: Ein existierendes Thing kann anhand des Namens oder der ID aus einer Liste ausgewählt werden

- Button "Neu": Existiert ein Thing noch nicht so kann es durch den Button "neuërstellt werden. Es öffnet sich eine Dialogbox, die es ermöglicht ein neues Thing zu erstellen
- Textfelder SSpalte": Hier kann durch Angabe der Spaltennummern eingestellt werden in welchen Spalten das Datum/die Uhrzeit zu finden ist
- Textfelder "Format": Hier wird per RegEx-String eingestellt, in welchem Format das Datum in den Spalten vorliegt
- Button "+": Hier können mehr Spalten, die das Datum beinhalten, ausgewählt werden
- "NameDropdown-Menü: Hier kann der (Multi-)Datastream über seinen Namen oder seiner ID aus einer Liste ausgewählt werden
- Spalten-Textfelder: Hier kann zu der vorangehenden Einheit die Spalte, die den Wert enthält, angegeben werden
- Button "Neu": Durch Klick auf den Button öffnet sich ein Dialogfenster, welches das Erstellen eines neuen (Multi-)Datastreams ermöglicht
- Button "+": Durch klick auf den Button "+"wird eine neue Box von Feldern erstellt, in der man Spalten auf Werte eines Datastreams abbilden kann
- Textfeld "Name": Hier kann der Name des neuen Things eingegeben werden. Dieser kann frei gewählt werden, darf aber nicht leer sein.
- Textfeld "Description": Hier kann eine Beschreibung des Thing in einem oder mehreren Worten eingegeben werden. Die Beschreibung kann auch ausgelassen werden.
- Textfeld "Property"(bsph. belegt durch property1): Hier kann der Name einer Property (siehe SensorThings API -¿ Things -¿ Properties) eingetragen werden
- Textfeld "Value"(bsph. belegt durch value1): Hier kann zur Property der Wert eingetragen werden. Werte "true"/"false"werden automatisch zu Boolean umgewandelt, Zahlenwerte zu Integer/Float und Textwerte zu Strings

- Button Ädd Property": Hier wird ein neues Textfeld-Paar Property/Value zum Interface hinzugefügt, damit mehrere Properties eingetrage werden können
- Dropdown-Menü "Location": Hier ann der derzeitge ort des Things aus einer Liste aus Orten ausgewählt

0.10 Produktdaten

Die gesicherten Daten der Software sind minimal, konkret heißt das es wird nur ein Log-File gesichert. Die Konfiguration wird dem Nutzer per CFG-Datei zum Download bereitgestellt und nicht Serverseitig gespeichert.
Sollte das Kriterium [Wunsch: Automatischer Download] mit aufgenommen werden, so muss die Adresse der entfernten Datei und des zu nutzenden FROST-Servers auf dem Server gespeichert werden.

- Log-Datei
- Abspeichern der Konfigurationen
- Speichern der Domain des FROST-Servers auf dem Server
- Speichern der Domain der entfernten CSV-Adresse auf dem Server

0.11 Produktleistungen

Die Software soll sämtliche Aktionen der Nutzer in einem Log-File speichern. Die Eingaben sollen möglichst direkt verarbeitet und die Reaktionszeit möglichst gering sein. Bei langsamen Operationen soll der Nutzer Feedback über eine Fortschrittsleiste und über das Log-Fenster bekommen.
Die Daten müssen korrekt an den FROST-Server übermittelt werden, dazu gehört auch Robustheit der Anwendung gegenüber falschen Angaben. Bei falschen Angaben oder sonstigen Fehlern soll der Nutzer mittels einer Fehlermeldung benachrichtigt werden.

- Logging
- Statusmeldungen an den Nutzer
- Schnelle Reaktionszeit auf Benutzer-Eingaben

- korrekte Übertragung der Daten
- Robustheit gegenüber falschen Formatangeben (Fehlermeldungen)

0.12 Nichtfunktionale Anforderungen

Die Software ist Open-Source und benötigt einen erreichbaren FROST-Server. Sie basiert auf der SensorThings-API, welche wie der FROST-Server ebenfalls Open-Source ist. Das Programm wird in Java geschrieben und hängt dementsprechend von einer Java-Umgebung ab. Um Sicherheit zu gewährleisten sollte das Programm robust gegenüber schädlichen Anfragen sein, vor allem zu große oder zu viele Anfragen sollten dementsprechend abgelehnt werden um eine Überlastung des Servers zu vermeiden. Außerdem sollten die daten HTTPS-/SSL-Verschlüsselt werden um Abfangen der daten zu verhindern.

(einzuhaltende Gesetze, Normen, Urheber- und Markenrechte, Sicherheitsanforderungen, Plattformabhängigkeiten)

- Genutzt wird der FROST-Server und die SensorThings API (beides Open-Source)
- Die Software ist Open-Source (GPLv3)
- Die Software ist abhängig von der Java-Plattform
- Robustheit gegenüber zu großen bzw. zu vielen Anfragen
- HTTPS-/SSL-Verschlüsselung

0.13 Qualitätsanforderungen

Das Programm sollte über lange Zeiträume auch ohne Unterbrechungen zuverlässig ausgeführt werden können, solange die Produktumgebung dies zulässt. Die Benutzbarkeit hat höchste Priorität, auch Nutzer ohne oder wenig technischem Vorwissen sollten in der Lage sein die Software zu bedienen. Außerdem soll es möglichst einfach sein die Software schnell und unkompliziert zu ändern oder zu erweitern. Eine einfache Portierung ist gegeben, da die Software auf Java basiert und damit Platformunabhängig ist. Es wird lediglich eine Produktumgebung vorausgesetzt die die Betriebsbedingungen erfüllt.

(In diesem Kapitel wird den Qualitätsmerkmalen Funktionalität, Zuverlässigkeit, Benutzbarkeit, Effizienz, Änderbarkeit und Übertragbarkeit je eine Qualitätsstufe aus sehr gut, gut, normal und nicht relevant zugeordnet.)

- Die Software sollte zuverlässig auch über längere Zeiträume ohne Unterbrechung laufen sofern es die Produktumgebung zulässt
- Die Benutzbarkeit hat höchste Priorität, es soll auch Nutzern mit wenig Computerkentnissen möglich sein die Software zu bedienen
- Es soll einfach möglich sein, Änderungen und Erweiterungen an der Software vorzunehmen
- Die Software ist einfach zu übertragen, da sie auf der plattformunabhängigen Programmiersprache Java basiert, es wird nur eine Produktumgebung vorausgesetzt, die lediglich die Betreibsbedingungen erfüllt

0.14 Anwendungsszenarien

Ideen für Anwendungsszenarien:

- Nutzer ohne technische Kenntnisse des Frost Server Aufbaus besitzt einen Frost Server und einen Sensor, der schon auf dem Frost Server abgespeichert ist. Außerdem hat er eine csv-Datei mit Daten vom Sensor, der einen schon bestehenden Datensatz des Sensors auf dem Server erweitern soll. Er navigiert in seinem Browser zur Web-oberfläche und stellt seine Datei als Quelle und den Frost Server als Ziel ein. Dann wählt er eine Favoritenkonfiguration aus, die zu seinem Sensor passt und lädt seine Daten durch Klick des ÜploadButtons hoch. Die Daten werden formatiert und auf dem Server abgrspeichert.
- das gleiche mit einem erfahrenen Nutzer, der ein neues Thing, einen Sensor und zugehörigen Datastream erstellt
- Szenario zum Abspeichern einer Konfiguration

0.15 Testfälle

Beschränkte Länge NICHT IMPLEMENTIERT **M1**
Kurz-URL Erstellen **T1**

Testet:

Testfall 1 [Grundlage] ; Stand : Offenes Browserfenster Aktion : Der Nutzer ruft die Website auf Reaktion: Die Weboberfläche wird angeziegt

Testfall 2 [Default-Case] ;

Stand: Radio Box Üpload File"wird angezeigt. Aktion: Button "Brwose"Reaktion: Systemdialog öffnet sich, csv-datei ausgewählt und hochgeladen

Stand : Datei ist ausgewählt und hochgeladen. Aktion : Der Nutzer gibt die Zieladresse (FROST-Server) an. Reaktion : Das Programm überprüft die Adresse und hinterlegt die Textbox grün(Funktioniert)/rot(Funktioniert nicht)

Stand : Grundlage Aktion : Der Nutzer drückt auf Upload. Reaktion : Das File wird hochgeladen.

Testfall 3 [CFG-File laden] ; Stand : Grundlage Aktion : Der Nutzer drückt auf "Browseünd wählt eine Config-Datei aus. Reaktion : Die Konfigurationen werden geladen.

Testfall 4 [Einstellen ohne Neues] ; Stand : Grundlage Aktion : Der Nutzer füllt die Textfelder SSpalte"für das Datum und die Zeit aus. Dann stellt er durch Klick auf den "+"Button eine zweite Spalte ein und füllt im Textfeld "Format"das RegEx für das Datum aus. Er wählt im Thing-Dropdown Menü ein Thing aus und fügt einen Data- und einen Multi- datastream hinzu. Beide werden ebenfalls durch ein Dropdown Menü ausgewählt. Nun wählt er die zugehörigen Spalten aus.

0.16 Zeitplanung

Beispielhafte Terminplanung:

- 07.05. - 27.05.: Pflichtenheft
- 28.05. - 24.06.: Entwurfsphase
- 25.06. - 22.07.: Implementierung
- 23.07. - 12.08.: Beispiel-Urlaubstermin (2 Wochen)
- 13.08. - 02.09.: Qualitätssicherung
- 03.09. - 09.09.: Terminfenster interne Abnahme
- 10.09. - 17.09.: Terminfenster Abschlusspräsentation

0.17 Glossar

- FROST
  Der **FR**auenhofer **O**penSource **S**ensor**T**hings-Server ist die erste komplette quelloffene Implementierung der OGC SensorThings API

- OGC
  Das **O**pen **G**eospatial **C**onsortium ist eine gemeinnützige Organisation, die sich zum Ziel gesetzt hat, die Entwicklung von raumbezogener Informationsverarbeitung (insbesondere Geodaten) auf Basis allgemeingültiger Standards festzulegen.

- SensorThings API
  Allgemeiner einheitlicher Standard um Verbindung zwischen Sensoren, deren Ort und Umgebung und den von ihnen erzeugten Daten herzustellen.

- CSV-/XLSX-Dateien
  CSV (Comma Seperated Value, .csv) und Excel (.xlsx) sind gängige Dateiformate zum Speichern von Daten (hier speziell: Sensor-Daten) in Tabellenform.

- CFG-Datei
  Konfigurationsdateien(kurz: CFG-Dateien) sind Textdateien, die speziell zur Speicherung bestimmter Einstellungen (Konfigurationen) dienen.

- Thing
  Ein Thing (deutsch: Ding) ist nach Definition der SensorThings API ein Objekt der physischen oder virtuellen Welt. Es benötigt einen Namen und eine Beschreibung. Es können sich ein oder mehrere Datastreams auf ein Thing beziehen.

- Datastream
  Ein Datastream ist eine fortlaufende Sammlung von Messungen (Observations) des selben Sensors.

- Integer
  Mit Integer wird in der Informatik ein Datentyp bezeichnet, der ganzzahlige Werte speichert.

- NULL
  Als Nullwert (kurz NULL) bezeichnet man in der Informatik einen Zustand, der das Fehlen eines Wertes anzeigen soll.

- Magic Numbers
  Deutsch: Magische Zahl. Ein im Quellcode eines Programms auftauchender Zahlenwert, dessen Bedeutung sich nicht unmittelbar erkennen lässt.

- Log-Datei
  Eine Log-Datei, auch Protokoll-Datei, enthält das automatisch geführte Protokoll aller oder bestimmter Aktionen von Prozessen auf einem Computersystem.

- Docker
  Docker ist eine Software zur Isolierung von Anwendungen. Docker vereinfacht die Bereitstellung von Anwendungen, weil sich sog. Container, die alle nötigen Pakete enthalten, leicht als Dateien transportieren und installieren lassen.

- GUI
  Ein Grafische Benutzeroberfläche (engl. **G**raphical **U**ser **I**nterface) hat die Aufgabe, eine Anwendungssoftware auf einem Rechner mittels grafischer Symbole und Steuerelemente für den Benutzer bedienbar zu machen.

- (Java) JRE
  Mit der Java-Laufzeitumgebung (**J**ava **R**untime **E**nvironment) werden Programme (Java-Anwendungen) weitgehend unabhängig vom darunter liegenden Betriebssystem ausgeführt.

- JavaScript
  JavaScript ist eine Programmiersprache, die hauptsächlich auf Webbrowsern Anwendung findet.

- headless Server
  Als headless (kopflos) bezeichnet man Computer, meist Serversysteme, die über keinen Bildschirm oder sonstige grafische Ausgabe verfügen. Diese Systeme haben als Steuerungsmöglichkeit in vielen Fällen eine Netzwerkverbindung.

- README
  Als README (deutsch: Liesmich) wird eine Datei bezeichnet, die üblicherweise mit Software mitgeliefert wird und diverse, oft wichtige Informationen über die Software enthält.

- Regex
  Regular **Ex**pressions (Regex, regulärer Ausdrucke) dienen der Beschreibung von Zeichenketten mit ähnlichem Format. Sie können als Filterkriterien in der Textsuche verwendet werden, indem der Text mit dem Muster des regulären Ausdrucks verglichen wird.

- HTTPS/SSL
  Hyper**T**ext **T**ransfer **P**rotocol **S**ecure (HTTPS) ist ein sicheres Hypertext-Übertragungsprotokoll, das SSL benutzt. SSL steht für **S**ecure **S**ockets **L**ayer und ist die Standardtechnologie für die Absicherung von Internetverbindungen und den Schutz sensibler Daten, die zwischen zwei Systemen übertragen werden.

- GPLv3
  GPLv3 ist die aktuelle Version von GPL. Diese **G**eneral **P**ublic **L**icense (allgemeine Veröffentlichungsgenehmigung) ist die am weitesten verbreitete Softwarelizenz, die einem gewährt, die Software auszuführen, zu studieren, zu ändern und zu verbreiten.