BACHELOR PAPER

Term paper submitted in partial fulfillment of the requirements for the degree of Bachelor of Science in Engineering at the University of Applied Sciences Technikum Wien - Degree Program Biomedical Engineering

A smart title
Really smart title

By: Gregor Beyerle
Student Number: 1210227035

Supervisor: FH-Prof. Dipl. Ing. Dr. Lars Mehnen

Vienna, May 9, 2015

FACHHOCHSCHULE TECHNIKUM WIEN

Declaration

“As author and creator of this work to hand, I confirm with my signature knowledge of the relevant copyright regulations governed by higher education acts (for example see §§21, 46 and 57 UrhG (Austrian copyright law) as amended as well as §11 of the Statute on Studies Act Provisions / Examination Regulations of the UAS Technikum Wien).

In particular I declare that I have made use of third-party content correctly, regardless what form it may have, and I am aware of any consequences I may face on the part of the degree program director if there should be evidence of missing autonomy and independence or evidence of any intent to fraudulently achieve a pass mark for this work (see §11 para. 1 Statute on Studies Act Provisions / Examination Regulations of the UAS Technikum Wien).

I further declare that up to this date I have not published the work to hand nor have I presented it to another examination board in the same or similar form. I affirm that the version submitted matches the version in the upload tool.“

Place, Date Signature

Kurzfassung

ICH BIN EIN TEST

Schlagworte: BOINSO

Abstract

I AM A TEST

Keywords: BOINSO

Acknowledgements

I acknowledge this paper as a piece of bullcr*p

Contents

1 Introduction

The use of pico and micro satellites in the context of educational aerospace programs has introduced opportunities for both private enthusiasts and students in small scale cost efficient satellite endeavors. Mostly lacking central oversight and costly infrastructure around the globe the efficiency of those programs is depending on the collaborative efforts of a growing community sharing both hardware, computational infrastructure and time. Due to the characteristics of orbiters deployed in a Low Earth Orbit (LEO) a stationary observer can establish a direct line of sight only up to twelve times a day – as stated in [3] – which can be maintained for about fifteen minutes per pass.

The challenges posed by the difficult monitoring of small satellites orbiting in LEO and possible ways to overcome them were outlined in [1]. This bachelor paper focuses on the implementation of an accessible and easily modifiable solution to connect Mission Control Centers (MCCs) and give Ground Control Centers (GCCs) the opportunity to deliver viable contributions to the monitoring process.

In the following chapters the reader is going to be presented with an outline of the tools and materials used to achieve this goal, the structure of the different applications written for this task as well as details about their implementation. Finally there will be a recapitulation of challenges which arose during the work on the project and an outlook on future developments.

1.1 BOINSO Core Web Application

The BOINSO Core Web Application is the main entry point for MCC administrators for maintaining the data related to their managed satellites. Besides the administrative interface the Core Web Application offers an Application Programming Interface (API) accessible via Representational State Transfer (REST) calls. An additional part of the Core Application is an OAuth2 provider which is used to authenticate users with a MCC.

1.2 BOINSO MCC Web Client

The BOINSO MCC Web Client is a HTML5/JavaScript sample implementation of a custom BOINSO client. It offers GCCs the possibility to register with a MCC, manually download Two Line Elements (TLE) files, administer their profile data and their tracking data.

1.3 BOINSO GPredict Bridge

The BOINSO GPredict Bridge is a piece of middle-ware distributed as a python module. It is used to pull updates of satellite data from the MCCs to their affiliated GCCs and in return automatically upload tracking data to the BOINSO Core Web Application of the satellites owner using the client API.

2 Material and Methods

The following sections list the different tools used while working on the components of the BOINSO network applications.

2.1 General

This section includes tools which allow a group of developers to work on complex projects in a decentralized manner. As this project is intended to be used in an academic context and monetary resources are scarce in this field tools that are either open source or free of charge for educational programs.

2.1.1 GIT-SCM

The Git Source Code Mirror (git-scm) is a "[...] fast, scalable, distributed revison control system with a [...] rich command set that provides both high-level operations and full access to internals" as declared in [2]. It follows the same "branch -> develop -> merge" work-flow as Subversion (SVN) and is freely distributed under the GNU General Public Licence Version 2 (GPLv2). It was originally developed by Linus Torvalds to offer Linux Kernel developers a free way to collaboratively work on a distributed code-base efficiently.

2.1.2 GitHub

GitHub is a provider of cloud hosted git-scm repositories. As GitHub was originally introduced by members of the open source community it still maintains a very generous relationship to open source contributors. GitHub users who publish their work to public repositories may use virtually all services bound to the GitHub infrastructure free of charge with only minimal limitations. GitHub also offers premium enterprise accounts which include a certain amount of private repositories and premium services if they are needed.

2.1.3 Travis-CI

Travis-CI is a web hosted continuous integration server. Continuous integration is the automated process of building and testing a project with every introduction or modification of a software component. Its goal is to assure and improve the quality of the project while alerting the development team if a change would lead to a breaking application.

In contrast of other continuous integration solutions like Jenkins – an open source continuous integration server implemented in the Java programming language – the configuration of a Travis-CI process is done by adding a simple configuration file to root of your git-scm repository as seen in listing 1.

```
language: python

python:
     - "2.7"
     - "3.4"

services: postgresql

before_script:
     - psql -c 'create database travisci;' -U postgres

install:
     - pip install -r requirements.txt
     - pip install coveralls

script:
     - coverage run --rcfile=.coverage.rc manage.py test

after_success:
     coveralls --rcfile=.coverage.rc
```

Code 1: BOINSO travis config file

2.1.4 Vagrant

Vagrant is a tool which is used to virtualize development environments. It offers a command line interface to download, start up, pause, halt and provision images of virtual machines which come configured with all the dependencies needed to develop, run and test an application. Base images can be built to closely model a production system as closely as possible being configured by a trained system administrator masking the complexity of this system from developers and designers. Depending on the base image type Vagrant users have to have access to a certain virtualization provider – also known as hypervisor.

The initialization process of a Vagrant environment depends on the presence of a vagrant configuration file – simply known as the Vagrantfile. Listing 2 depicts the Vagrantfile of the BOINSO Core Web Application.

```
Vagrant.configure(2) do |config|

  config.vm.box = "WalternativE/django_base_box"
  config.vm.provision :shell, path: "bootstrap.sh"
  config.vm.network :forwarded_port, host: 8000, guest: 8000

end
```

Code 2: BOINSO Core Web Application Vagrantfile including expressions to set the virtual base box, a simple shell provisioner executing a bash script which installs variable project dependencies, and the automated port forwarding from the virtual box to the host development machine

Provisioning can be done by DevOps tools like Puppet or Chef or any other program used in this context. As the scope and the resources of this project are limited a simple shell provisioner was used. An example for a provisioning script can be seen in listing 3.

```
#!/usr/bin/env bash

cd /vagrant
sudo rm /var/lib/apt/lists/* -R
sudo apt-get update
sudo apt-get upgrade -y

# whatever you want to provision comes here
# right now only shell provisioning works in this box
# chef/puppet will be added (eventually)

sudo pip3 install -r requirements.txt

echo "provisioner is done"
exit 0
```

Code 3: Vagrant shell provisioning bash script

2.2 BOINSO Core Web Application

This section focuses on the tools and frameworks used to implement the core of the BOINSO Core Web Application.

2.2.1 Pip

In a modern Python environment pip is used to manage Python modules. Pip itself is a python module which accesses the Python Package Index (PyPI) downloads a module and its dependencies, starts the compilation process for Python extensions and either adds the module to the global Python interpreters Python path or the path of a local virtual Python environment. Besides installing

Bibliography

[1] BEYERLE, G.: *An open source solution for distributed ground and mission control communication for low earth orbit satellites – A feasibility study*, 2014.

[2] GIT-SCM USER GROUP: *Git - the stupid content tracker*, 2014.

[3] KIEF, C., R. BUFFINGTON, N. PURUSHOTHAM, R. S. ERWIN, J. ANDROLEWICZ, J. LYKE and J. JACKSON: *GENSO, SPA, SDR and GNU Radio: The Pathway Ahead for Space Dial Tone*. Infotech@Aerospace, 2011, 2011.

List of Figures

List of Tables

List of Code

List of Abbreviations

API	Application Programming Interface
GENSO	Global Educational Network for Satellite Operations
BOINC	Berkeley Open Infrastructure for Network Computing
LEO	Low Earth Orbit
GEO	Geostationary Orbit
GPL	GNU General Public License
MCC	Mission Control Center
GCC	Ground Control Center
COTS	Commercial off-the-shelf
ISEB	International Space Education Board
CSA	Canadian Space Agency
ESA	European Space Agency
JAXA	Japan Aerospace Exploration Agency
NASA	National Aeronautics and Space Administration
CPU	Central Processing Unit
GPU	Graphics Processing Unit
PRC	Public Resource Computing
TNC	Terminal Node Controller
LTS	Long Term Support
URL	Uniform Resource Locator
CGI	Common Gateway Interface
MPL-2.0	Mozilla Public License Version 2.0

NORAD	North American Aerospace Defense Command
HamLib	Ham Radio Control Libraries
TLE	Two Line Elements
git-scm	Git Source Code Mirror
SVN	Subversion
GPLv2	GNU General Public Licence Version 2
REST	Representational State Transfer
PyPI	Python Package Index