



Informatik an der OST

Im Fokus: Open Source und Open Data

Bulletin

1/2022

Prof. Stefan Richter
Studiengangleiter
Informatik



Als vor rund 40 Jahren die Pioniere der Free- and-Open-Source-Bewegung anfangen, Software zu veröffentlichen, deren Quellen allen Interessierten zugänglich waren und beliebig an eigene spezifische Bedürfnisse angepasst werden konnten, wurden sie kaum ernstgenommen. In den 1990ern und 2000ern empfanden einige Unternehmen die Bewegung hingegen als Bedrohung für ihre Geschäftstätigkeit; der Ton war entsprechend gehässig. Mittlerweile haben sich die Gemüter etwas abgekühlt und Open-Source-Software hat sich in einigen Bereichen gar als Standard etablieren können. So sind zum Beispiel Linux als Betriebssystem und die GNU Compiler Collection als C und C++ Compiler führend. Auch Softwareunternehmen haben den Wert von Open-Source-Software-Projekten erkannt und tragen aktiv zur Weiterentwicklung bei. Inzwischen kann man davon ausgehen, dass diese Koexistenz von kommerzieller und quelloffener Software weiter Bestand haben wird, auch weil sich die Geschäftsmodelle in den letzten Jahrzehnten stark gewandelt haben und durch Open-Source-Software sehr viel mehr Menschen am technischen Fortschritt teilhaben können. An der OST unterrichten wir sowohl an und mit Open Source sowie proprietärer Software, um unseren Studierenden ein möglichst breites Bild der heutigen Informatik zu vermitteln.

Save the Date:

Am Freitagnachmittag, 29. April 2022, 14–18 Uhr (bzw. 21 Uhr), ist am OST-Campus Rapperswil-Jona ein Marathon geplant unter dem Motto «Meine Umwelt – mein Klima». Der Event ist Teil der von Studierenden organisierten Nachhaltigkeitswoche Rapperswil. <https://www.sustainabilityweek.ch/rapperswil/>

Die Prinzipien von Open Source und Open Data gehen uns alle etwas an

Was ist Open Source?

Open Source ist Software, deren Quellcode alle frei einsehen und verändern dürfen. Ihren Ursprung hat sie in der Hacker-Bewegung der Sechzigerjahre sowie in frühen Softwareentwicklungen an Hochschulen. Kaum jemand weiss, dass 1982 am damaligen Technikum Rapperswil – heute OST Campus Rapperswil-Jona – die Unix Interessen-Gemeinschaft Schweiz (UNIGS) von Leuten, die von offenen Systemen begeistert waren, gegründet wurde. Daraus ist der Dachverband CH Open entstanden, mit dem der Autor heute noch die damals entstandenen Workshop-Tage durchführt.

Open Source ist allgegenwärtig

Open Source ist überall zu finden: im Betriebssystem Android, mit dem die meisten Smartphones ausgestattet sind, in den Autos von Tesla, BMW oder Mercedes, in der internationalen

Raumstation – und vor allem in den Webinfrastrukturen, in der Cloud und im Internet der Dinge.

Obwohl Open-Source-Programme kostenlos sind, lässt sich damit Geld verdienen: durch Wartung, Support und Anpassung. Immer wieder ist auch Strategie im Spiel. So kann sich eine Firma einen Vorteil versprechen, indem sie ihre Software in Bereichen offenlegt, wo sie Defizite hat. Oder aber indem eine Firma dadurch der Konkurrenz die Margen drückt. Eine weitere Motivation kann sein, dass eine Firma damit Ideen von Freiwilligen einholt, wie das zum Teil mit Softwarebibliotheken für die Künstliche Intelligenz geschieht (siehe auch Fokus Künstliche Intelligenz im Bulletin 2/2021). Wie bedeutend Open Source ist, zeigt zum Beispiel der Milliardenbetrag, den IBM 2018 bezahlte, um die Open-Source-Firma Red Hat zu übernehmen.

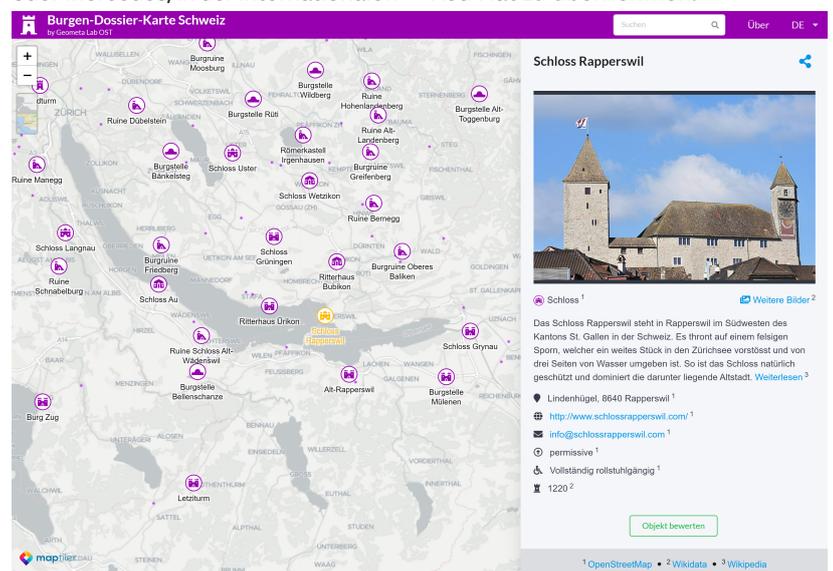


Abbildung: Die fünfssprachige Burgen-Dossier-Karte Schweiz und Umgebung vom Institut für Software mit Daten von Wikipedia, Wikidata und OpenStreetMap. Die Webapp wurde durchgehend mit Open-Source-Software entwickelt und ist selber auch als Open Source verfügbar. <https://castle-map.infs.ch/> (eigene Abbildung)

Open Source weitergedacht

Die Idee von Open Source geht aber noch weiter. Sie besagt, dass man ohne Genehmigung oder Registrierung Teil eines technischen Netzwerks sein kann, mithilfe föderierter Systeme. Jeder verwaltet seine eigenen Daten. Dazu braucht es Standards. Ein gutes Beispiel dafür ist das E-Mail: Man muss nicht wissen, welches Programm der andere benutzt. Das Einzige, was man kennen muss, ist seine Mail-Adresse. Das liegt daran, dass E-Mail auf Standards und föderierten Technologien aufbaut. Dass das Prinzip in den höchsten Etagen der Bundesverwaltung angekommen ist, zeigt die App SwissCovid, die mit Open-Source-Komponenten und mit föderierter Architektur realisiert wurde.

Open Source wird unterschätzt

Die digitale Welt enthält viel mehr Freiwilligenarbeit, als man denkt. Das thematisierte ein Artikel der NZZ im Dezember 2021 zur «übersehenen Bedeutung von Open Source», nachdem in einer Java-Bibliothek (log4j) eine weltweit bedrohliche Sicherheitslücke entdeckt worden war. Diese Lücke wurde von zwei Ingenieuren – darunter ein Schweizer – in ihrer Freizeit geschlossen.

Open Source wird bedroht

Open Source wird in letzter Zeit aber auch bedroht, v. a. durch Grosskonzerne. Zum Beispiel kann man sich auf zahlreichen Websites auch mit anderen Logins einloggen. Das ist dank eines offenen Login-Standards (OAuth) möglich. Nun haben Facebook, Microsoft und Google diese Funktion bei sich aber abgestellt, so dass das Login nur noch mit dem eigenen Konto funktioniert. Amazon (AWS) und Alibaba Web Services nutzen sehr viel Open Source und verdienen damit viel Geld. Jedoch geben sie nur einen Bruchteil wieder zurück an die Gemeinschaft.

Was ist Open Data?

Als Open Data werden Daten bezeichnet, die von allen zu jedem Zweck genutzt und weiterverwendet werden dürfen. Open Data von Behörden werden Open Government Data genannt und diese verbreiten sich immer mehr. Open Data kann aber auch von Freiwilligen stammen, so dass Wissensdatenbanken entstehen wie Wikipedia, Wikidata oder OpenStreetMap (OSM). OSM ist eine Google-Maps-Alternative. Empfehlenswert ist beispielsweise das Offline-Navi-App Organic Maps.

Was geht uns das an?

Für junge Informatiker*innen sind Open Source und Open Data eine Gelegenheit, ihr Können zu belegen. Für Betreuende sind sie ein Beitrag zur langfristigen und breiten Zugänglichkeit des digitalen Wissens.

Ich meine, dass man grundsätzlich mehr auf Open Source und Open Data sowie auf Standards und föderierte Systeme achten sollte. Und vielleicht ergibt sich ja die Gelegenheit, sich selber zu engagieren – zum Beispiel an einem kommenden Hackathon oder Mapathon an der OST.

(KEST) 

Hörtipp:

Prof. Stefan F. Keller gibt im SRF-Podcast im Kapitel «Die Libraries, die das Internet zusammenhalten» Auskunft über das Thema Open-Source-Sicherheit.

Der Podcast kann hier angehört werden: <https://www.srf.ch/audio/digital-podcast> > Achtung-fertig-Homeoffice 

pytest: Automatisiertes Testen von Software

Automatisiertes Testen ist ein essenzielles Werkzeug für qualitativen Code. Ohne ein Werkzeug für die einfache Erstellung von solchen Tests stösst man bei der Entwicklung von Programmen schnell an die Grenzen: Manuelles Testen der Funktionalität ist aufwändig und fehleranfällig.

Schnell kommt man an einen Punkt, an dem das Vertrauen in den eigenen (oder eben oft fremden) Code sinkt: Man wagt es kaum noch, Änderungen am Code vorzunehmen, ohne die Angst, dabei versehentlich etwas «kaputt zu machen». In der Python-Welt wird oft [pytest][1] benutzt, um solche Tests zu implementieren. Dies bestätigt eine [Umfrage][2] von JetBrains zusammen mit der Python Software Foundation: Von den über 28'000 Teilnehmenden nutzen 49 % pytest, jedoch nur 28 % das mit Python mitgelieferte «unittest».

37 % geben an, gar kein Test-Framework zu nutzen. Es wäre schön, wenn diese Zahl in den Umfrageergebnissen für 2021 kleiner würde. Diese erscheinen vermutlich etwa zeitgleich mit diesem Newsletter.

Welche Vorteile bietet pytest also gegenüber den Python-Boardmitteln?

Das Framework nutzt eine deutlich einfachere Syntax für das Verfassen von Tests. Dies ermöglicht es, diese sowohl übersichtlicher als auch unkomplizierter umzusetzen, als dies mit dem unittest-Modul möglich wäre. Auch die Ausgabe bei Fehlern ist besser gestaltet, was die Entwicklung vereinfacht. Der Wechsel zu pytest für existierende Projekte ist nahtlos möglich, da pytest auch bestehende, für unittest verfasste Tests ausführen kann. Weiterhin lässt es sich problemlos an die eigenen Ansprüche anpassen und mit diversen Plug-ins erweitern. Obwohl pytest primär auf Python ausgelegt ist, eignet es sich auch gut als Basis für andere Systeme: So wird es etwa auch benutzt, um IoT-Kühlschränke, Waschmaschinen,

Automobil-Komponenten oder gar Audio-Codecs wie MP3 zu testen.

Auch an der OST wird pytest vielseitig genutzt und gelehrt: Ob als Fundament für Netzwerk-Tests mit dem NUTS-Projekt, als Teil von Projekten im Modul «Automatisierung mit Python» oder in Firmenkursen zu Python.

Mit Florian Bruhin (dem Autor dieses Artikels) ist gar einer der Hauptentwickler von pytest an der OST präsent.

[1]: <https://pytest.org/>

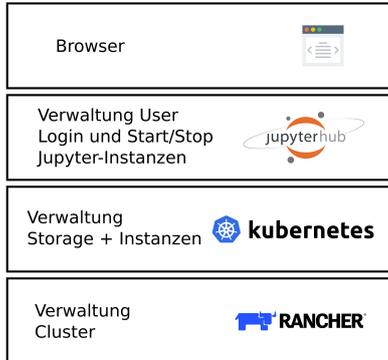
[2]: <https://www.jetbrains.com/lp/python-developers-survey-2020/>

(BRFL) 



Moderne Übungen dank Open-Source-Software

Alles geht in die Cloud. Informatik-Unterricht mit Übungen bisher nicht. Wo das erwünscht ist, braucht es eine einheitliche (webbasierte) Lösung. Am Institut für Software haben wir das testweise mithilfe von Open-Source-Software umgesetzt.



Unser Stack von unten nach oben: Wir betreiben mit Rancher (Cluster-Verwaltung) einen Kubernetes-Cluster (sprich: «eine Cloud»). Darauf läuft die Software-Applikation JupyterHub, welche sich einerseits um das Login kümmert und andererseits auch die übungsspezifischen Umgebungen startet und stoppt. Sowohl die Cluster-Verwaltung (Kubernetes mit Rancher) als auch der JupyterHub sind vollständig freie Softwares und somit lizenzkostenfrei erhältlich und nutzbar. Sie brauchen vor der Nutzung jedoch eine individuelle Anpassung, die mit gewissen Herausforderungen

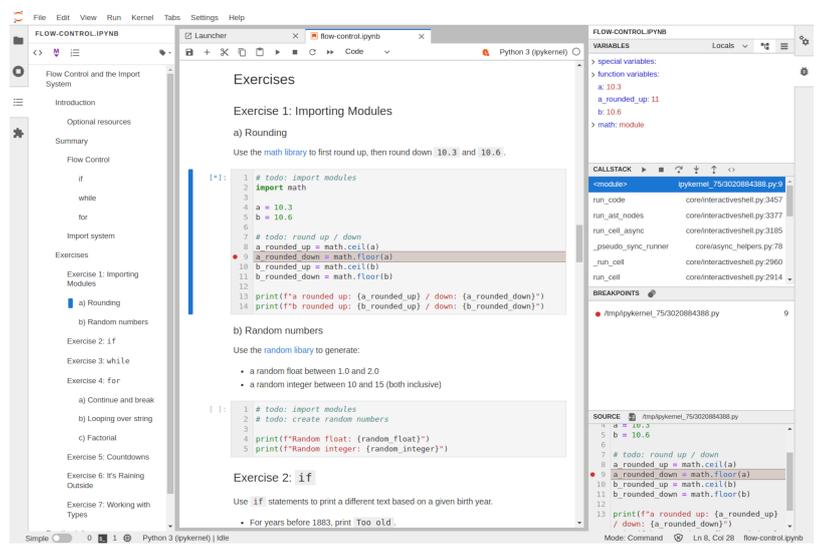
verbunden ist. Warum machen wir das denn nicht ohne Cluster?

Die aufwändigere Kombination von JupyterHub mit Kubernetes erlaubt uns, sogenannte Begleit-Container zu starten, welche dann zum Beispiel eine Datenbank oder einen Abgabeserver beinhalten. So stellen wir eine hohe Flexibilität sicher, um auf die übungsspezifischen Bedürfnisse einzugehen.

Die Jupyter-Umgebung bietet eine relativ einfache Programmier-Umgebung und bietet Unterstützung für verschiedenste Sprachen. Da finden sich unter anderem Python, Haskell,

Julia und SQL. Auch ausserhalb des Unterrichtsumfeldes – vor allem in den Bereichen Big Data, Künstliche Intelligenz und Data Science – erfreut sich Jupyter grosser Beliebtheit. In naher Zukunft werden wir das System um einzelne GPUs erweitern. Längerfristig planen wir mehr Flexibilität mit einer modernen Programmierumgebung wie serverseitigem Visual Studio Code. Weiter stellen wir uns vor, gewisse Abgaben für Testate oder Übungen in einer externen E-Learning-Umgebung anzuzeigen.

(JONI)



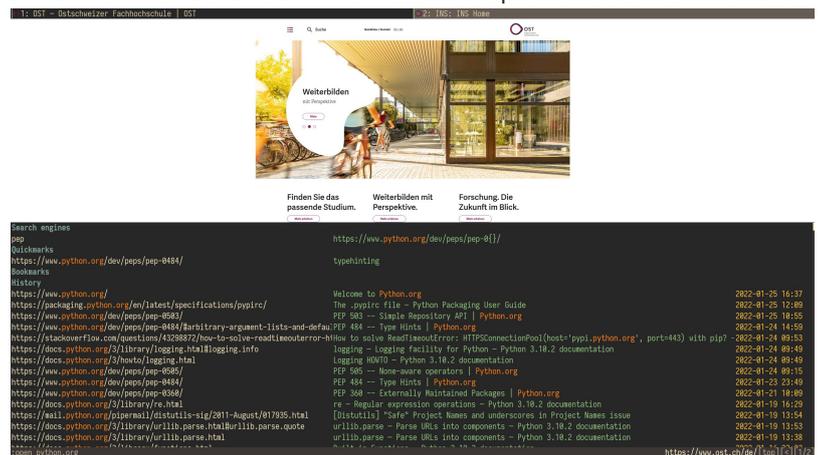
qutebrowser: Ein Webbrowser für Tastatur-Liebhaber*innen

Gängige Webbrowser wie Google Chrome oder Mozilla Firefox sind typischerweise primär auf die Bedienung mit der Maus ausgelegt: Es existieren gewisse Tastaturkürzel für eine schnellere Bedienung, aber eine Maus ist unabdingbar. Weiterhin liegt der primäre Fokus dieser Projekte – verständlicherweise – auf einer einfachen Bedienung durch Personen ohne eine spezielle Affinität zu Computern.

Das [qutebrowser][1]-Projekt verfolgt einen anderen Ansatz: einen Browser zu entwickeln, der zwar intern auf denselben Grundbausteinen wie Google Chrome aufbaut, jedoch eine eher ungewöhnliche Oberfläche dafür bietet. Diese soll möglichst minimal sein und der gesamte Browser soll effizient mit der Tastatur bedienbar sein. Die Bedienung ist dabei inspiriert vom Konzept des vi-Editors, der ursprünglich 1976 herausgegeben wurde und (mit der Weiterentwicklung «vim») bis heute eine rege Beliebtheit etwa unter Software-Entwickler*innen und Systemadministrator*innen geniesst.

Daraus ergibt sich, dass sich qutebrowser eher auf eine Nische der

«Power-User» fokussiert: Personen also, die ihr System präferiert mit der Tastatur bedienen und typischerweise eher «exotische» Ansprüche an ihre Arbeitsabläufe bzw. Programme pflegen. Durch die Entwicklung als «kompletter» Browser anstatt einer Erweiterung für Firefox oder Chrome bietet qutebrowser eine viel nahtlose-



re Integration in solche Umgebungen, als es vergleichbare Erweiterungen wie Vimium oder Tridactyl können.

Das Projekt wurde 2013 von Florian Bruhin lanciert, der an der OST das Modul «Automatisierung mit Python» leitet. Auch der Browser ist

grösstenteils in Python geschrieben. 2019 wurde das Projekt im Rahmen einer Studienarbeit an der OST um die Möglichkeit ergänzt, eigene einfache Python-Erweiterungen dafür zu entwickeln. Der Browser ist weiterhin in reger Entwicklung, dank einer erfolgreichen langfristigen Spenden-

finanzierung der Nutzer*innen.

[1]: <https://qutebrowser.org/>
[2]: <https://eprints.ost.ch/id/eprint/743/>

(BRFL) 

NUTS – Network Unit Testing System

Computernetze sind das IT-Rückgrat jedes Unternehmens. In Zeiten grosser Umbrüche in der IT sind auch diese Netzwerke starken Veränderungen unterworfen und durch die zunehmende Automatisierung nimmt gleichzeitig die Anzahl Änderungen am Netzwerk zu.

Doch wie wird sichergestellt, dass das Netzwerk auch nach den Änderungen weiterhin wie gewünscht funktioniert und dass eine Änderung keine unerwünschten Effekte auf das

Netzwerk hat? Genau um solchen Herausforderungen zu begegnen, hat sich in der Software-Entwicklung seit Jahrzehnten automatisiertes Testen bewährt. Dieses Prinzip kann natürlich auch auf Netzwerke übertragen werden, zum Beispiel in Form der Software «NUTS» (Network Unit Testing System), welche am INS Institut für Netzwerksysteme der OST – Ostschweizer Fachhochschule entwickelt wird.

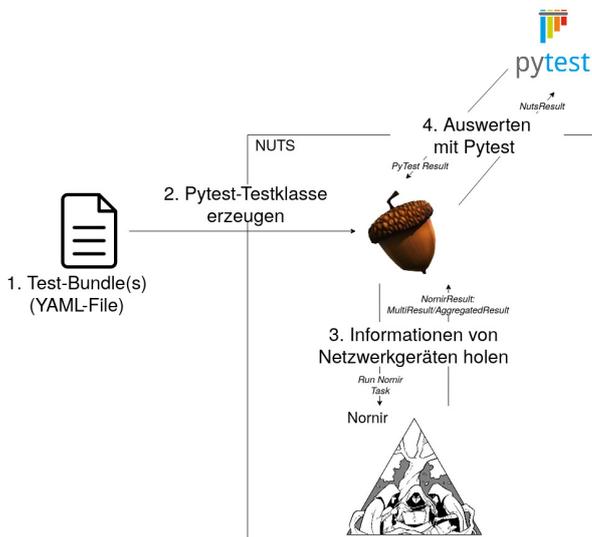
NUTS erlaubt es, ein Netzwerk in verschiedenen Facetten zu testen, von der Konfiguration einzelner Netzwerk-Ports bis hin zu Durchsatzttests zwischen verschiedenen Netzwerksegmenten – und das jederzeit auf «Knopfdruck». Die gewünschten Tests werden Netzwerkadministratorfreundlich als deklarative YAML-Dateien definiert und anschliessend von

NUTS herstellerunabhängig auf den entsprechenden Netzwerkgeräten durchgeführt. Soeben ist NUTS in der Version 3.1 erschienen, welche insbesondere das Testportfolio erweitert.

Unter der Haube verwendet NUTS das Netzwerkkonfigurations-Framework Nornir, welches wiederum auf den Frameworks Napalm und Netmiko zur Unterstützung von Netzwerkgeräten verschiedener Hersteller aufbaut. NUTS ist als pytest-Plugin konzipiert und profitiert damit von den mächtigen und etablierten Test-Auswertungs-Features von pytest. Sowohl NUTS wie auch die erwähnten Frameworks sind Open-Source-Softwares.

NUTS kann mit einem einfachen «pip install nuts» installiert werden, der Quellcode von NUTS ist unter <https://github.com/INSRapperswil/nuts> gehostet und die Dokumentation ist auf <https://nuts.readthedocs.io> zu finden.

(MURL) 



Registration is open for ZuriHac 2022!



After spending the last two years as a virtual event, ZuriHac 2022 is planned to take place as a physical event from Saturday, June 11 to Monday, June 13, 2022 at the Rapperswil-Jona campus of the OST! ZuriHac is the biggest practitioner's conference in the world centered around the

functional programming language Haskell. It features invited talks from leading experts from industry and academia, and hands-on beginners and advanced tracks.

Aside from the talks and tracks, attendees are encouraged to work to-

gether on all sorts of Haskell-related open-source projects and socialize.

Registration is free and open on <https://zurihac.info>

(MEFA) 

Impressum

Redaktion
Studiengang Informatik, Claudia Furrer (FUCL),
<https://www.ost.ch/i>, Tel. 058 257 49 09

Autoren
Prof. Stefan Richter (RSTE), Prof. Stefan Keller (KEST), Prof.
Dr. Farhad Mehta (MEFA), Florian Bruhin (BRFL), Nicola
Jordan (JONI), Lukas Murer (MURL)

