

SDDC

Software Defined Data Center

Studienarbeit

Hochschule für Technik Rapperswil
Institute for Networked Solutions

Herbstsemester 2015

Autoren:	Silvan Adrian, Fabian Binna
Betreuender Dozent:	Prof. Beat Stettler
Betreuer:	Urs Baumann
Gegenleser:	TBA
Experte:	TBA
Projektpartner:	Institute for Networked Solutions

Abstract

TODO

Erklärung der Eigenständigkeit

Ich erkläre hiermit,

- dass ich die vorliegende Arbeit selber und ohne fremde Hilfe durchgeführt habe, ausser derjenigen, welche explizit in der Aufgabenstellung erwähnt ist oder mit dem Betreuer schriftlich vereinbart wurde,
- dass ich sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben habe.
- dass ich keine durch Copyright geschützten Materialien (z.B. Bilder) in dieser Arbeit in unerlaubter Weise genutzt habe.

Ort, Datum:

Raperswil-Jona, 9. Dezember 2015

Name, Unterschrift

A handwritten signature in black ink, appearing to read 'S. Adrian', with a horizontal line drawn underneath the name.

Silvan Adrian

Ich erkläre hiermit,

- dass ich die vorliegende Arbeit selber und ohne fremde Hilfe durchgeführt habe, ausser derjenigen, welche explizit in der Aufgabenstellung erwähnt ist oder mit dem Betreuer schriftlich vereinbart wurde,
- dass ich sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben habe.
- dass ich keine durch Copyright geschützten Materialien (z.B. Bilder) in dieser Arbeit in unerlaubter Weise genutzt habe.

Ort, Datum:

Raperswil-Jona, 9. Dezember 2015

Name, Unterschrift

Fabian Binna

Inhaltsverzeichnis

Abstract	1
1 Einleitung	6
1.1 Aufgabenstellung	6
1.2 Vorbemerkungen	6
1.3 Zweck	6
I Technischer Bericht	7
2 Analyse	8
2.1 Ist-Situation	8
2.2 Soll-Situation	8
2.3 API Analyse	9
2.4 Support	10
2.5 Fazit	13
2.6 Bitnami Analyse	18
2.7 Fazit	30
2.8 Dashboard Analyse	31
2.9 Security	37
2.10 Fazit	38
2.11 Akteure	39
2.12 Use Case Diagramm	40
2.13 User Stories Skizzen	40
2.14 Rollen	40
2.15 Ziele	41
2.16 Epic	41
2.17 User Stories	41
2.18 Domainmodell Skizze	43
3 Anforderungen	44
3.1 API	44
3.2 Customer-Dashboard	45

3.3	Admin-Dashboard	46
3.4	Use Cases	47
3.5	Epics	55
3.6	User Stories	55
3.7	Nicht-funktionale Anforderungen	61
4	Design	63
4.1	REST API	63
4.2	CRUD Service	63
4.3	CRUD OrderedService	64
4.4	Generic API	64
4.5	Operationen	64
4.6	Configfile	68
4.7	Architektur	70
4.8	Systemübersicht	70
4.9	Logische Architektur	70
4.10	Klassenstruktur	72
4.11	Dependency Injection	74
4.12	Deployment	75
4.13	Persistierung	75
4.14	Sequenzdiagramme	76
5	Appendix	77
6	Glossar	78
7	Abkürzungsverzeichnis	79

1 Einleitung

1.1 Aufgabenstellung

Unter "Software Defined" versteht man die Zentralisierung der Intelligenz in Controllern. Gerade moderne Data Center werden immer mehr von Software Controllern gesteuert, damit die Dynamik der Bereitstellung von neuen Services massiv erhöht werden kann. So gibt es bereits Controller für Storage, Netzwerk und Compute Ressourcen.

Ziel ist es, die Ressourcen Storage, Network und Compute abstrahiert als skalierbare Pools der SService Ebene zu Verfügung zu stellen. Alle modernen Controller können über API's angesprochen werden, allerdings unterscheiden sich hier die verschiedenen Hersteller zum Teil stark. Ziel dieser Arbeit ist die Entwicklung einer generischen Middleware/API, um verschiedene Controller möglichst einfach in Business Applikationen zu integrieren. Nach der Definition einer systemunabhängigen Schnittstelle sollen die verschiedenen Controller danach als Treiber an die API angehängt werden können. Zur Demonstrationszwecken soll eine rudimentäre, ca. 3 Seitige Webpage erstellt werden, welche die erstellte API benutzt. Dabei soll je mind. ein Storage, Compute und Network Controller eingebunden werden.

1.2 Vorbemerkungen

TODO

1.3 Zweck

Diese Arbeit soll die Möglichkeiten aufzeigen, um Software Defined Data Center (SDDC) in möglichst einfacher und generischer Art zu betreiben. TODO--

Teil I

Technischer Bericht

2 Analyse

2.1 Ist-Situation

2.1.1 Anbieter

Der mit Abstand grösste Anbieter

2.1.2 Libraries

2.2 Soll-Situation

2.2.1 API

- Die API sollte auf einem möglichst stabilen Stand sein.
- Es müssen die wichtigsten Provider zur Verfügung stehen.
- Die API muss gut dokumentiert sein.
- Es sollen verschiedene Services angesprochen werden können (Compute, Storage, Network...).
- Keine grosse Einarbeitung, das heisst die Programmiersprache sollte nicht komplett neu sein.

Zudem sind alle Eigenschaften die das Implementieren der Software erleichtern ein Pluspunkt. Von Vorteil wären zusätzliche Funktionen wie z.B SSL oder Pricing.

2.2.2 User-Dashboard

Das User-Dashboard soll eine Möglichkeit für Benutzer bieten, um einzelne Services abonnieren zu können. Dabei soll sowohl IaaS, PaaS oder SaaS abonniert werden können und eine Auswahl bieten unter vielen verschiedenen Cloud Anbietern wählen zu können (so generisch wie möglich). Dabei soll der User zwischen einzelnen Angeboten der Anbieter spezifischen Services zu wählen bspw.: bei Google Cloud: Cloud DNS, Firewall, Netzwerke etc. Es kann daher auch sein das nicht alle Anbieter die gleichen Services bieten und daher eine Auswahl gegeben werden muss, damit der Benutzer selbst entscheiden kann welchen Service er haben will.

SDDC

Unser Projekt soll deshalb eine einiger generische Möglichkeit bieten, um Service abonnieren zu können und wenn möglich so gut wie alle Cloud Anbieter zu unterstützen. Dies soll möglich werden indem ein Dashboard eine generische API anspricht und die API alle Schritte durchführt, die nötig sind für die Erstellung des Services.

2.2.3 Admin-Dashboard

Dem Admin soll eine Möglichkeit geboten werden um die Software administrieren zu können, z.B.: Benutzerverwaltung oder etwas in der Art.

2.3 API Analyse

2.3.1 Libcloud¹

Sprache: Python

Wichtigste Provider: Rackspace, Amazon web services, CloudStack, OpenStack, DigitalOcean, Eucalyptus, Joyent, Linode, exoscale, NephoScale, Google Cloud Platform, Zerigo, CloudSigma, iKoula, libvirt

2.3.2 jClouds²

Sprache: Java

Wichtigste Provider: OpenStack, Docker, DigitalOcean, Google Cloud Platform, Rackspace, HP Cloud, CloudStack, Amazon web services, abiquo, CloudSigma, joyent

2.3.3 elibcloud³

Sprache: Erlang

elibcloud ist ein Wrapper für libcloud.

2.3.4 fog⁴

Sprache: Ruby

Wichtigste Provider: CloudSigma, CloudStack, GoGrid, Google Cloud Platform, Joyent, Libvirt, Linode, OpenStack, OpenVZ, Rackspace, Zerigo, IBM, HP

¹Python library for interacting with many of the popular cloud service providers using a unified API. URL: <https://libcloud.apache.org/>.

²The Java Multi-Cloud Toolkit. URL: <https://jclouds.apache.org/>.

³Erlang wrapper around Libcloud. URL: <https://github.com/esl/elibcloud>.

⁴The Ruby cloud services library. URL: <https://github.com/fog/fog/>.

2.3.5 pkgcloud⁵

Sprache: JavaScript (Node.js)

Wichtigste Provider: Amazon, Azure, DigitalOcean, Joyent, OpenStack, Rackspace, Google, HP

2.3.6 libvirt⁶

Sprache: C

Wichtigste Provider: Xen, KVM, OpenVZ, VMware ESX, VirtualBox, IBM PowerVM

2.4 Support

2.4.1 Compute

Die grösste Auswahl an Providern liefert Libcloud.JClouds hingegen unterstützt auch Docker, was ein grosser Vorteil gegenüber Libcloud ist. Im Abschnitt 2.5.6 wird genau aufgeführt, welche Provider von welchen APIs unterstützt werden. Es werden nur public Clouds berücksichtigt. Für den private Cloud Bereich bietet sich hier eher Libvirt an, da neben XEN, KVM, Qemu und weitere unterstützt werden.

2.4.2 Storage (Object/Blob)

libcloud

- PCextreme AuroraObjects
- Microsoft Azure (blobs)
- CloudFiles
- Google Storage
- KTUCloud Storage
- Numbus.io
- Ninefold
- OpenStack Swift
- Amazon

jclouds (BlobStore)

- AWS
- HP Helion
- Azure
- Rackspace

fog

⁵*pkgcloud is a standard library for node.js that abstracts away differences among multiple cloud providers.* URL: <https://github.com/pkgcloud/pkgcloud>.

⁶*libvirt: The virtualization API.* URL: <http://libvirt.org/>.

- S3
- CloudFiles

pkgcloud

- Amazon
- Azure
- Google

- Google Storage

- HP
- OpenStack
- Rackspace

libvirt

- GlusterFS
- Sheepdog
- SCSI
- iSCSI

- FiberChannel
- NFS
- lvm
- filesystems

2.4.3 Network

Libvirt allein bietet hier mit Abstand die beste Unterstützung von Network Konfigurationen (VLANs, Bridges, etc.) und ist sehr auf Private Clouds ausgelegt.

2.4.4 Other

Database

pkgcloud

- IrisCouch
- MongoLab
- Rackspace
- MongoHQ
- RedisToGo

DNS

libcloud

- AuroraDNS
- DigitalOcean
- Gandi
- Google
- Host Virtual
- Linode

- Rackspace
- AWS Route53

fog

- AWS Route53
- Blue Box
- DNSimple
- Linode

pkgcloud

- Rackspace

Load Balancer

libcloud

- Brightbox
- CloudStack
- DimensionData
- Amazon
- Google

jclouds

- AWS Elastic LoadBalancer

pkgcloud

- Rackspace

Orchestration

pkgcloud (beta)

- OpenStack

CDN

fog

- CloudFront

- Softlayer
- Zerigo

- Rackspace
- Rage4
- Slicehost
- Zerigo

- GoGrid
- Ninefold
- Rackspace
- Softlayer

- Rackspace

- Rackspace

2.5 Fazit

Im Gesamtbild schneidet libcloud am besten ab. Es bietet deutlich am meisten Compute und Storage Provider. Die Dokumentation ist sehr ausführlich, mit konkreten Ratschlägen zur Implementation (z.B. Thread Safe). Zusätzlich bietet libcloud Module für SSL und Pricing. Jclouds ist eine Library für Java, was für uns am besten ist, da wir am meisten Erfahrung mit Java haben. Es gibt jedoch nicht viele Compute Provider, dafür unterstützt jclouds als einziger Docker. Der einzige Vorteil von fog ist die Möglichkeit CDNs als Service anzubieten. Pkgcloud unterstützt eine breite Auswahl von Services (z.B. Database, Load Balancer, DNS). Elibcloud ist ein erlang Wrapper für libcloud und unterstützt somit das gleiche wie libcloud (sonlange die Version auf dem neusten stand ist). Erlang würde sich für eine parallele Umgebung eignen, es sind jedoch keinerlei Erlang Kenntnisse im Team vorhanden.

Libvirt ist allerdings die einzige API, die sich völlig auf Private Anbieter zugeschnitten ist (wird auch von OpenStack verwendet), wäre also die Beste Lösung, wenn es darum geht eine Private Cloud aufzubauen.

Wir entscheiden uns für libvirt. Die Gründe dafür sind im Abschnitt 2.5.7 aufgeführt

2.5.1 libcloud

- ⊕ Grösste Auswahl an Compute und Storage Provider.
- ⊕ Am besten dokumentiert. Für jede Methode existiert eine Tabelle, die zeigt welche Provider damit angesprochen werden können.
- ⊕ Ist zwar nicht Thread-Safe. Es werden jedoch konkrete Lösungsvorschläge gemacht.
- ⊕ SSL und Pricing Module vorhanden.
- ⊖ Team hat wenig Erfahrung mit komplexen/grossen Python Projekten.

2.5.2 jclouds

- ⊕ Unterstützt Docker.
- ⊕ Java Library. Das Team hat am meisten Erfahrung mit Java.
- ⊕ Code Examples für fast jeden Provider.
- ⊖ Kleine Auswahl an Compute Providern.

2.5.3 fog

- ⊕ Es ist möglich ein CDN als Service anzubieten.
- ⊖ Mässige Dokumentation. Es existieren zwar Examples, die sind aber nicht besonders aussagekräftig.
- ⊖ Kleine Auswahl an Compute Providern.

2.5.4 pkgcloud

- ⊕ Grösste Auswahl an Services.
- ⊕ Database as a Service
- ⊕ Orchestration
- ⊕ Explizite Unterstützung von Network.
- ⊖ Mässige Dokumentation. Es existieren zwar Examples, die sind aber nicht besonders aussagekräftig.
- ⊖ Kleine Auswahl an Compute Providern.

2.5.5 libvirt

- ⊕ Grösste Auswahl an "Private Cloud Anbietern".
- ⊕ Grosse Storage Unterstützung.
- ⊕ Explizite Unterstützung von Network.
- ⊕ Bietet Java Library (language Bindings)
- ⊖ Ungenügende Dokumentation. Es existieren zwar Examples und sonst existiert nur eine Javadoc zur Library.

2.5.6 Compute Vergleich

	Libcloud	jClouds	fog	pkgcloud	libvirt
AWS					
Abiquo					
PcextremeAuroraCompute					
Azure					
Blue Blocks					
Brightbox					
CloudFrames					

CloudSigma					
CloudStack					
Cloudwatt					
DigitalOcean					
Docker					
DimensionData					
Dreamhost					
Enomaly Elastic Computing Platform					
ElasticHosts					
Eucalyptus					
Exoscale					
Gandi					
Go2Cloud					
Google Compute Engine					
GoGrid					
HostVirtual					
HP Public Cloud (helion)					
IBM SmartCloud Enterprise					
Ikoula					
Joyent					
Kili Public Cloud					
KTU Public Cloud					
Libvirt					
Linode					
NephoScale					
Nimbus					
Ninefold					
OpenHosting					
OnApp					
OpenNebula					
OpenStack					
Opsource					
Outscale					
Packet					
ProfitBricks					
Rackspace Cloud					
RimuHosting					
RunAbove					
ServerLove					
skalicloud					
SoftLayer					

vCloud					
VCL					
Voxel VoxCLOUD					
vps.net					
Vmware vSphere					
Vultr					
XEN					
KVM					
OpenVZ					
IBM PowerVM					
Hyper-V					
Virtuozzo					
Bhyve					
VMware ESX					
VirtualBox					

2.5.7 API Vergleichsmatrix

	Entwicklungsstand	Provider	Doku	Serviceauswahl	Sprache	Zusatz Features
libcloud	v0.18.0: Die API ist stabil und wird von namhaften Firmen verwendet.	* 35 Unterstützt die wichtigsten Provider	gut	Wenige Services, dafür umfangreiche Funktionen	Python	SSL, Pricing, Übertragen von Files, Ausführungen von Scripts, Unit Testability, Mocks
jclouds	v1.9.1: Die API ist stabil und wird von namhaften Firmen verwendet.	* 15 Unterstützt die wichtigsten Provider und als einziger Docker	genügend	Wenige Services, dafür umfangreiche Funktionen	Java	Thread-Safe, Unit Testability, Dateübertragung, Scriptausführung
fog	v1.34.0: Die API ist stabil, wird jedoch von keinen namhaften Firmen verwendet	sehr kleine Auswahl	ungenügend	Wenige Services. Unterstützt als einziger CDNs.	Ruby	
pkgcloud	V1.2.0: Wird von keinen namhaften Firmen verwendet. Teilweise Betastand.	sehr kleine Auswahl, die wichtigsten sind jedoch dabei.	ungenügend	Breite Auswahl	JavaScript (Node.js)	Unit Testability
Libvirt	1.2.20: Bietet eine stabile API in C (bietet auch Language Bindings für Java und Python.)	Grosse Auswahl für Private Anbieter (Xen, KVM etc.)	genügend	Grosser Umfang schön unterteilt auf Computer (Domain), Storage, Network	C	Private Anbieter Unterstützung, Thread-Safe, bietet Mock zum testen
	gut	*Verschiedene Standorte vom gleichen Provider zusammengefasst				
	genügend					
	ungenügend					

2.6 Bitnami Analyse

2.6.1 Einführung

Bitnami bietet ein Dashboard für einige Cloud Anbieter (VMware, AWS, Google Cloud, Azure, Digitalocean), um ganz einfach vorgegebene viel verwendete WebApps, Datenbanken oder Technologie Stacks schnell in der Cloud zu starten. Dabei wird bei jedem eine Compute Instanz erstellt und das Image installiert. Diese Analyse soll dabei helfen eine gute Lösung für unser Dashboard zu konzipieren und auf bereits bewährtes zurückgreifen können.

2.6.2 Cloud Anbieter

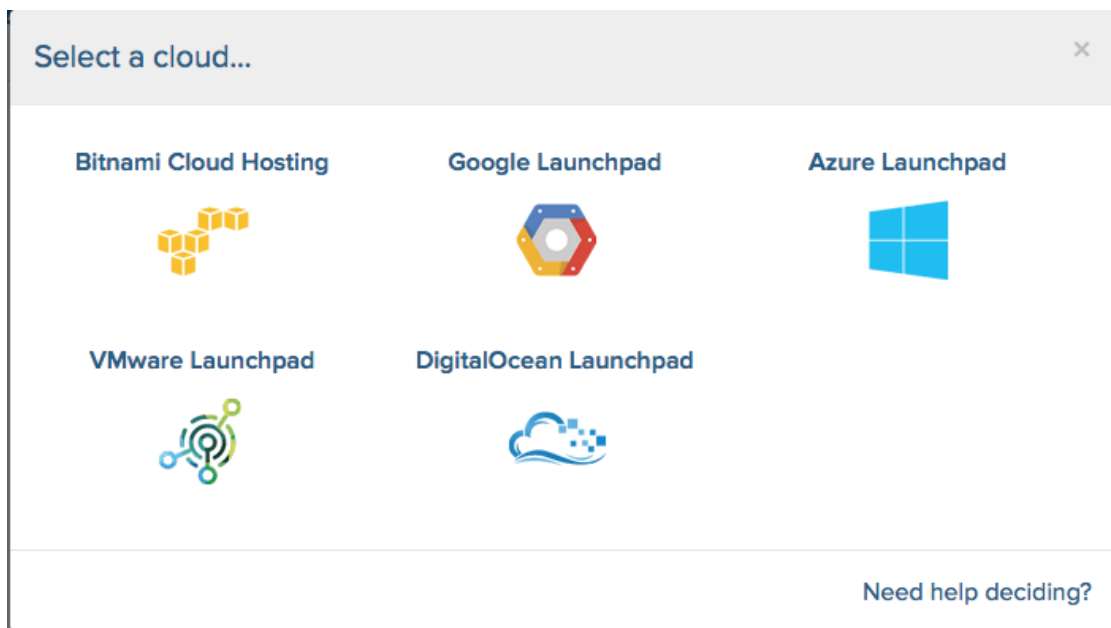


Abbildung 2.1: Cloud Anbieter

Bitnami Cloud Hosting⁷

Beim Bitnami Cloud Hosting steckt AWS dahinter, hier sind die meisten Einstellungen möglich im Vergleich zu den anderen Dashboards.

Übersicht

kurze Übersicht über die möglichen Konfigurationen einer AWS Instanz.

The screenshot displays the Bitnami Cloud Hosting configuration interface. At the top, there are input fields for 'Name' (test123) and 'Domain Name' (new-server-648ac3.bitnamiapp.com). Below this, the interface is divided into three main sections: Application Options, Development Options, and Add New Application. The Application Options section includes Ubuntu 14.04.1 64-bit, T1 Micro (\$14.40 /mo), and 10 GB (\$1.00 /mo). The Development Options section includes 1-Hour Demo, US East Coast (Virginia), and Dynamic IP. To the right, a box titled 'Estimated Amazon charges:' shows a total of \$15.40 /mo, with a note that the server may be free if you qualify. At the bottom right, there are buttons for 'Launch 1h Demo' and 'Cancel'.

Category	Option	Price /mo
Application Options	Ubuntu 14.04.1 64-bit	
	T1 Micro	\$14.40
	10 GB	\$1.00
Development Options	1-Hour Demo	
	US East Coast (Virginia)	
	Dynamic IP	

Estimated Amazon charges:

Option	Price /mo
T1 Micro	\$14.40
10 GB	\$1.00
Total	\$15.40

✓ Free Tier Eligible
Note that your server may be FREE if you qualify. [Learn more](#)
AWS charges may be incurred.

[Launch 1h Demo](#)
[Cancel](#)

Abbildung 2.2: AWS Übersicht

⁷Bitnami Launchpad for AWS. URL: <https://aws.bitnami.com>.

Applikationseinstellungen:

(a) AWS Application Options

Applikationsauswahl:

(b) AWS Applikationsauswahl

Abbildung 2.3: AWS Applikationseinstellungen

Betriebssystem:

(a) AWS Betriebssystem

Servertyp:

(b) AWS Servertyp

Abbildung 2.4: AWS Betriebssystem und Servertyp

Diskgrösse:

(a) AWS Diskgrösse

Region,IP,Account:

(b) AWS Region,IP und Account

Abbildung 2.5: AWS Diskgrösse + Region,IP,Account

Management:

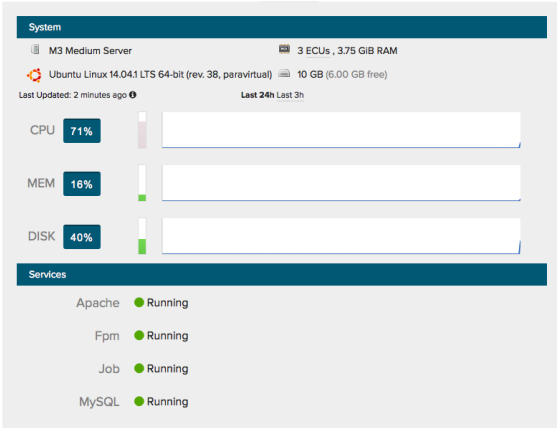
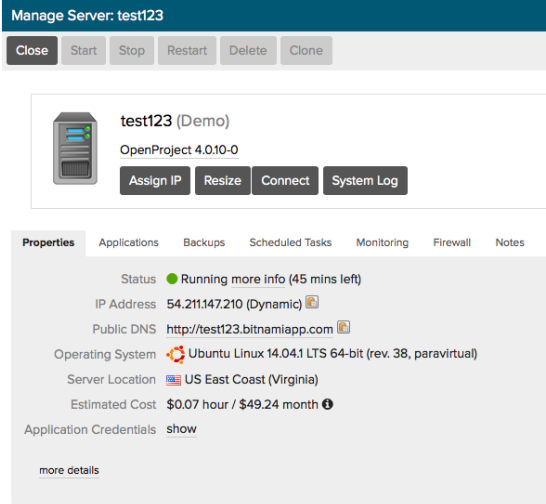
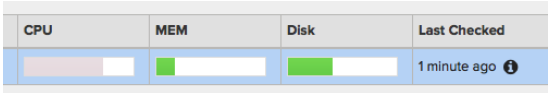


Abbildung 2.6: AWS Management Ressourcen

Ressourcen:



(a) AWS Server Management



(b) AWS Server Ressourcen

Abbildung 2.7: AWS Ressourcen

Digitalocean Launchpad⁸

Bitnami Library

Popular open source images, ready to launch on [DigitalOcean](#) in one click.

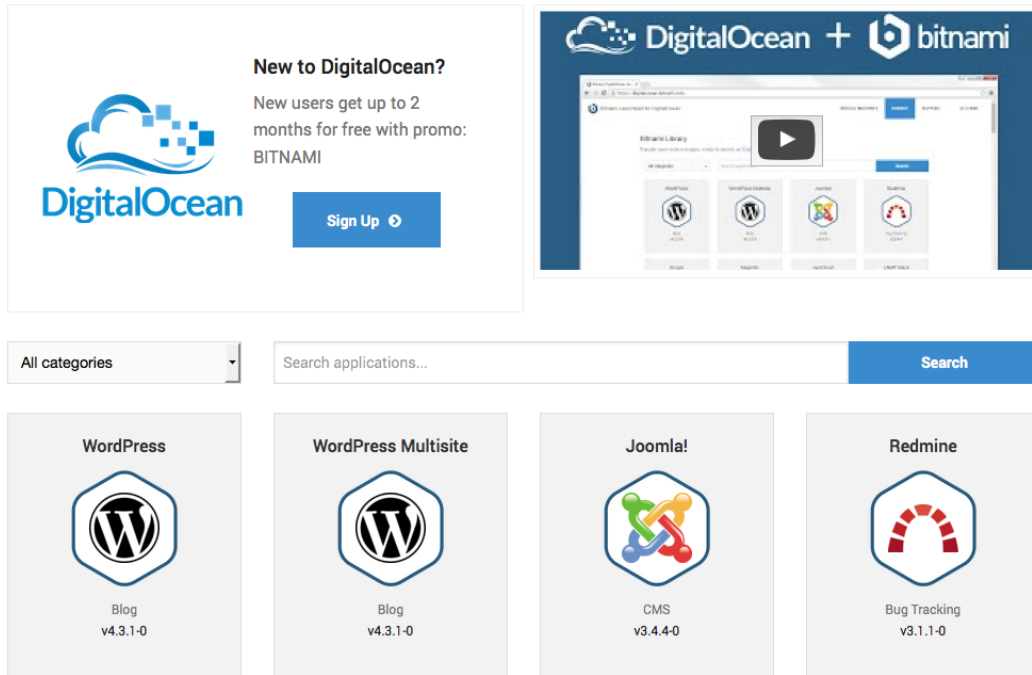


Abbildung 2.8: Digitalocean Launchpad

⁸Bitnami Launchpad for DigitalOcean. URL: <https://digitalocean.bitnami.com>.

Authorize Application:

Um Instanz erstellen zu können muss das Bitnami Dashboard Zugriff auf den Cloud spezifischen Account haben.

Authorize Application

Bitnami Launchpad would like permission to access your account: **sadrian@hsr.ch**

Review Permissions

- Read
- Write

Authorize Application

Deny

Bitnami Launchpad

Bitnami Launchpad for DigitalOcean

[Visit application website](#)

Abbildung 2.9: Digitalocean Applikation autorisieren

Instanzen:

Sobald eine Applikation ausgewählt wurde kann eine Instanz mit dem App gestartet werden, dabei kann noch die Instanzgrösse und Location gewählt werden.

New Virtual Machine

NAME: mywordpress-server

IMAGE: WordPress Multisite v4.3.1-0 (dod.debian-v4 v8.1)

CLOUD: Add cloud account

SERVER SIZE: 512MB (\$5.00 /mo) \$0.007 /hr 1 vCPU, 512MB memory, 20GB storage and 1.0TB data transfer

REGION: Amsterdam 3

NETWORK: Enable private networking, Enable IPv6

BACKUPS: Enable backups (+20% \$/mo)

Cancel Create

(a) Digitalocean Instanz

Instanzinfos: Sobald Instanz erstellt lässt sich deren Status überprüfen + App spezifische Links werden gesetzt und Private Key/Public Key lassen sich herunterladen.

test123

Running

Manage in the DigitalOcean Console

Application Info

WordPress 4.3.1-0

GO TO APPLICATION

GO TO ADMINISTRATION

GO TO CREDENTIALS

CREDENTIALS

USERNAME: user

PASSWORD: [masked]

Droplet Info

46.101.195.201

512MB \$5.00 /mo (\$0.007 /hr) 20 GB

SOLID STATE

FRA1 REGION

\$5.00 ESTIMATED MONTHLY COST

DOWNLOAD KEY: SSH, PEM, PPK

Show Less

Backups: disabled

Private networking: disabled

Public IPv6: disabled

Restart Shutdown Delete

(b) Digitalocean Instanzinfos

Abbildung 2.10: Digitalocean Instanz

Cloud Accounts:

Es können in Bitnami mehrere Cloud Accounts hinzugefügt werden

Your Cloud Accounts

DigitalOcean

Name	Virtual Machines	Options
1 hour demo (demo)	1	+ Launch VM
sadrian@hsr.ch (70b9eea607)	0	+ Launch VM x Delete

Add cloud account

Abbildung 2.11: Digitalocean Cloud Accounts

Und unter jedem Account können spezifisch VM's erstellt werden:

CLOUD [+ Add cloud account](#)

sadrian@hsr.ch (70b9eea607) ▼

Abbildung 2.12: Digitalocean Account spezifische VMs

Danach taucht die Instanz in der Übersicht auf:

Your Virtual Machines

Name	Application	Region	Cloud	Status
test123	WordPress	nyc3	1 hour demo	Downloading application ↻

New Virtual Machine

Abbildung 2.13: Digitalocean Instanzen

Azure Launchpad⁹

Beim Azure Launchpad wird wieder gleich vorgegangen, wie bei Digitalocean. Nur das sich die infos ändern (Bspw.: Bei Digitalocean Droplet, jetzt Server).

Ebenfalls ändern sich die Instanzgrößen. welche bei Azure anders als bei Digitalocean sind + wird bei Azure mit Subscriptions und nicht anhand von Accounts unterschieden.

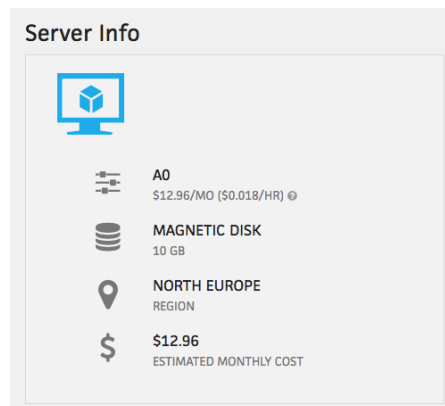
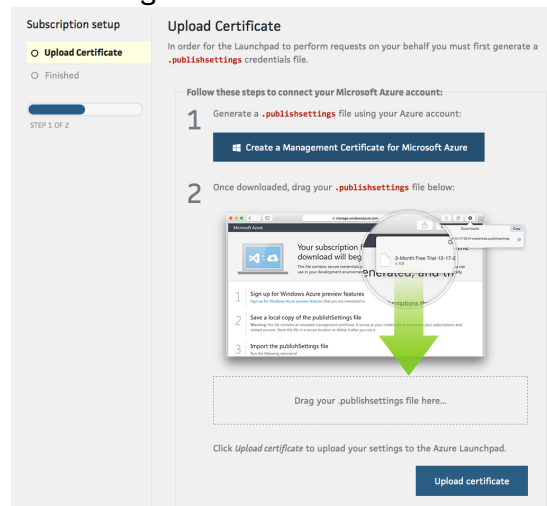


Abbildung 2.14: Azure Serverinfo

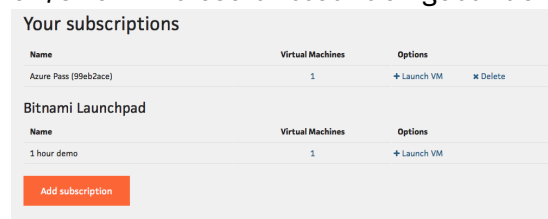
Authorization:

Bei Azure wird das Dashboard über ein Management Certificate autorisiert:



(a) Azure Management Zertifikat

Danach wird die vorhandene Subscription/s vom Microsoft Account eingebunden:



(b) Azure Subscription

Abbildung 2.15: Azure Authorization

⁹Bitnami Launchpad for Azure. URL: <https://azure.bitnami.com>.


Sobald dann ein neuer Server erstellt wurde kann dieser auch wieder gelöscht werden und all dessen Infos angesehen werden.

test123

Running


Manage in the Azure Console

Application Info




WordPress 4.3.1-0

WordPress is one of the world's most popular web publishing platforms for building blogs and websites. It can be customized...
[Learn More](#)




GO TO APPLICATION

LAUNCHES IN A NEW WINDOW



GO TO ADMINISTRATION

LAUNCHES IN A NEW WINDOW




CREDENTIALS

USERNAMEuser

PASSWORD*****


show

Server Info




40.127.180.163

bitnami-wordpress-1d5a.cloudapp.net




A0

\$12.96/MO (\$0.018/HR) ©




MAGNETIC DISK

10 GB




NORTH EUROPE

REGION



\$12.96

ESTIMATED MONTHLY COST



SSH Credentials

USERNAMEbitnamil

PASSWORD*****

show

ShutdownDelete

Abbildung 2.16: Azure Instanzinfos

26

Google Launchpad¹⁰

Beim Google Launchpad wieder dasselbe, wie bei Azure oder Digitalocean. Hier wird einfach mit Projekten unterschieden (schliesslich kann jedem Projekt mehrere Compute Instanzen oder andere Services angefügt sein.)

NAME

test123

PROJECT [Add project](#)

1 hour demo

You have 10 out of 10 demo server launches left.

NETWORK

default

DISK TYPE [?](#)

☐ Solid State ☒ Magnetic Disk

Disk size

100 GB

\$4.00 /mo

SERVER SIZE [?](#)

☒ **f1-micro** [?](#)
(\$4.54 /mo) \$0.009 /hr

☐ **g1-small** [?](#)
(\$15.12 /mo) \$0.030 /hr

☐ **n1-standard-1** [?](#)
(\$27.72 /mo) \$0.055 /hr

Estimated Monthly cost: **\$8.54** [?](#)

CANCEL **CREATE**

Abbildung 2.17: Google Instanzerstellung

¹⁰Bitnami Launchpad for Google. URL: <https://google.bitnami.com>.

VMware Launchpad¹¹

Das VMware Launchpad kann für die VMware vCloud Air gebraucht werden.

New Virtual Machine

NAME

test123

REGION

de-germany-1-15

VIRTUAL DATA CENTER (VDC)

vdC2

NETWORK

default-routed-network

SERVER PROPERTIES

CPU

1 CPUs

\$10.84

Memory

2 GB

\$32.47

Disk size

20 GB

\$2.93

Disk type

Standard

SSD Accelerated

Public IP Address

N/A

vCloud Air OnDemand Online Support (7%)

\$3.46

Estimated total cost

\$49.47 /mo

WordPress v4.3.1-0 (vmvcloudair-x64 v14.04)

WordPress is one of the world's most popular web publishing platforms for building blogs and websites. It can be customized via a wide selection of themes, extensions and plug-ins.

Cancel

Create

Infos:

Server Info

SSD ACCELERATED

20 GB (\$1.46/MO)

2 GB

MEMORY (\$32.47/MO)

1

CPU (\$10.84/MO)

DE-GERMANY

REGION

VDC2

VIRTUAL DATA CENTER (VDC)

\$80.35

ESTIMATED MONTHLY COST

NOT AVAILABLE

Show More

(a) VMware Instanzerstellung

(b) VMware Infos


Abbildung 2.18: VMware Launchpad

2.6.3 Security

Da durch das zentrale zusammenfassen von mehreren Accounts auch immer Sicherheitsrisiken zu beachten sind, wird bei Bitnami zusätzlich zum normalen Loginpasswort auch noch ein Vaultpasswort festgelegt, welches wohl die Logindaten symmetrisch verschlüsselt und in einem "Vault" ablegt.

¹¹Bitnami Launchpad for VMware. URL: <https://vmware.bitnami.com>.

Setup Your Bitnami Vault



Before continuing, we ask that you setup a password for your Bitnami Vault.

This password is independent from your DigitalOcean account and primary Bitnami account, and is used to secure access to sensitive information such as SSH keys or API credentials .

We can't recover it for you, so please be sure to write it down.

Vault password

Vault password confirmation

Save Password

Abbildung 2.19: Bitnami Security

2.6.4 Applikationen

Die Applikationen können sich von Anbieter zu Anbieter unterscheiden, jedoch gibt es für sehr viel verwendete Applikationen (Bspw.: Wordpress) bei jedem ein Image. Es besteht bereits eine sehr grosse Auswahl für sehr viel verschiedene Apps und es werden immer mehr, wodurch es immer einfacher wird schnell eine Applikation aufzusetzen.

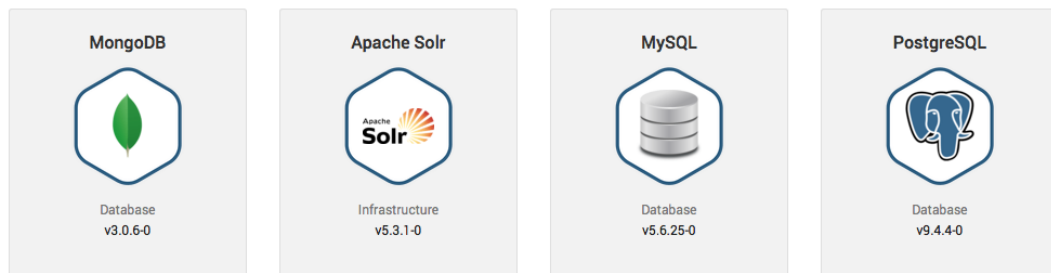


Abbildung 2.20: Applikationen

2.7 Fazit

Bitnami bietet einiges was Computing angeht und ist der einzig grössere Dashboard Anbieter, der mehrere verschiedene Cloud Anbieter unterstützt. Allerdings fehlen verschiedene PaaS Angebote (Bspw.: Cloud SQL bei Google etc.), Bitnami ist daher nur für eigene Instanzen/Vms zu gebrauchen und nicht mit einer generellen Service Unterstützung konzipiert worden. Das Dashboard bei Bitnami bietet jedenfalls einiges und gibt einem einen guten Überblick über seine eigenen abonnierten Services (VM Instanzen).

2.8 Dashboard Analyse

Das Dashboard soll dem Nutzer schnell und einfach die Übersicht über seine eigenen abonnierten Services bieten (Compute,Storage,Network). Dabei soll auf eine Anzahl von Cloud Anbietern angeboten werden, sowohl Public Cloud(z.B.: AWS, Google Cloud, Azure, Digitalocean), wie auch Private Cloud (z.B.:CloudStack.Open Stack). Die nachfolgenden Mockups sollen bereits einen ersten Eindruck der möglichen Funktionalitäten des Dashboard geben.

Beim Login wird zwischen Nutzern und Administrator unterschieden.

2.8.1 Homescreen Nutzer

Der Nutzer kriegt eine Übersicht über die vorhandenen Cloud Anbieter und kann gemäss Wunsch den richtigen wählen.

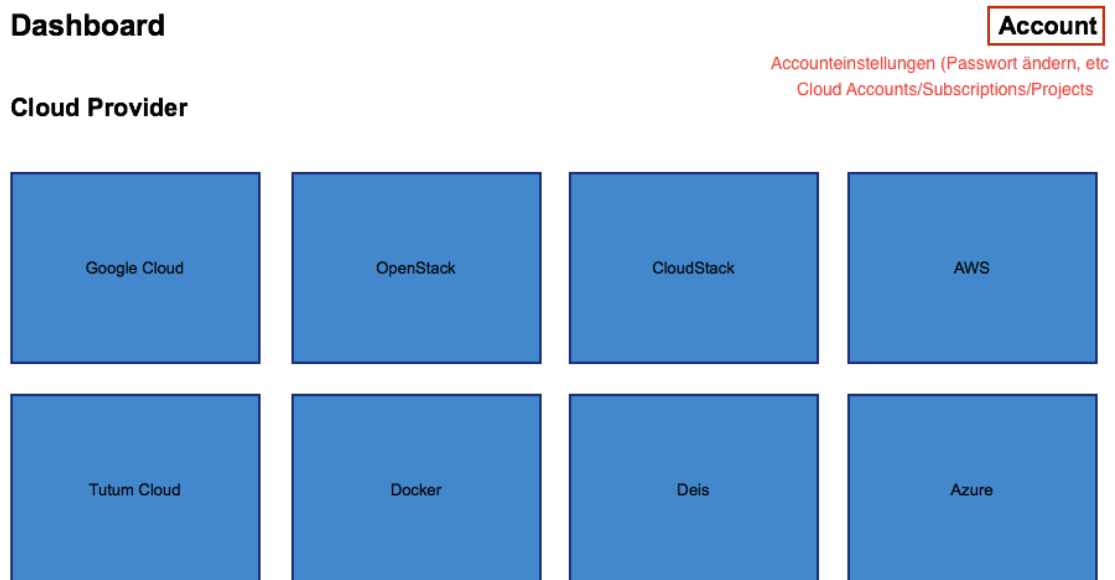


Abbildung 2.21: Homescreen Nutzer

2.8.2 Homescree Admin

Der Administrator kriegt lediglich eine Übersicht über die vorhanden Nutzer und kann diese löschen oder ändern.

Dashboard		Account
Users		
Name	Rolle	Aktion
silvan	Benutzer	löschen / ändern
fabian	Benutzer	löschen / ändern
Admin	Administrator	ändern
<button>create User</button>		

Abbildung 2.22: Homescree Admin

2.8.3 Offerings

Sobald man einer der Cloud Anbieter gewählt hat (auf dem Homescree) öffnet sich das jeweilige Dashboard des Anbieters mit dessen spezifischen Services.

Dashboard

Abonnierte Services **Services** Offerings Account

[Home](#) > [Google Cloud Services](#) [Cloud Service Angebote](#)

Google Cloud Offerings

Kategorien

Cloud SQL

Compute Engine

Cloud Datastore

Load Balancer

Firewall

VPN

Cloud DNS

App Engine

Abbildung 2.23: Homescree Google

Compute:

Hier werden nur Compute Offerings angezeigt z.B.: App Engine, Compute Engine, Container Engine, EC2 etc., können nach Anbieter variieren

Dashboard

Services Offerings Account

Google Cloud Services

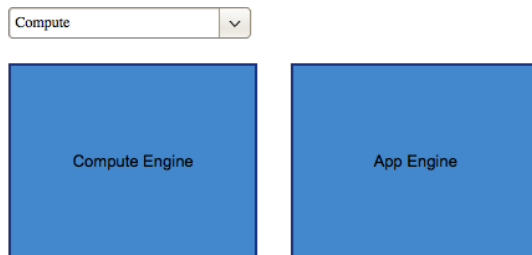


Abbildung 2.24: Homescreen Kategorie Compute

Storage:

Nur Storage spezifische Offerings anzeigen z.B.: Cloud Datastore, Cloud SQL, Cloud BigTable), die sich je nach Anbieter ändern.

Dashboard

Services Offerings Account

Google Cloud Services

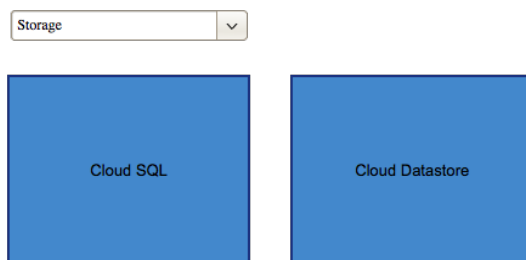


Abbildung 2.25: Homescreen Kategorie Storage

Network:

Network spezifische Offerings anbieten (Firewall, VPN, Netzwerke, Cloud DNS etc.) und dann verändert sich die Auswahl auch Anbieter spezifisch.

Dashboard

Services Offerings Account

Google Cloud Services

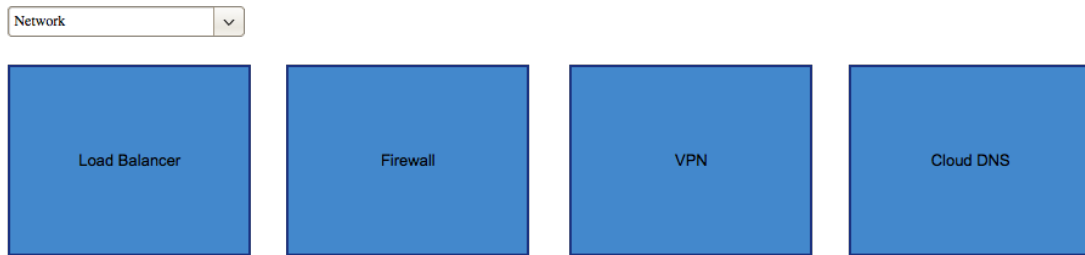


Abbildung 2.26: Homescreen Kategorie Network

2.8.4 Services

In Services kriegt man einen Überblick über all seine abonnierten Services des jeweiligen Anbieters und kann den zu bearbeitenden wählen.

Dashboard

Services Offerings Account

Your Services

Name	Service	Category	Region	Project	Options
WebApp Database	Cloud SQL	Storage	europe-west1-b	My test Project	Active
OpenProject Server	Compute Engine	Compute	europe-west1-b	My test Project	Running
Firewall test Project	Firewall	Compute	-	My test Project	Active

[new Service](#)

Abbildung 2.27: Services Übersicht

Compute

Wenn man einen Compute Service wählt kriegt man eine Übersicht des Services, welcher Storage, Leistung + Region (Storage und Compute werden jedoch meistens vorgegeben durch die Instanzgrößen) + werden die Kosten pro Monat angezeigt. Im Management kann dann die Instanz direkt neugestartet/heruntergefahren oder gelöscht werden + sollen noch Links (Ip etc.) zur Verfügung stehen um sich auf die Instanz verbinden zu können.

Dashboard

Services Offerings Account

OpenProject Server

Info
F1_micro
Disk-Size
10GB
Region
EUROPE-WEST1-B
Costs
5\$ / Month

Management
[delete](#) [restart](#) [shutdown](#)
Links
<https://console.google.com/ComputeEngine>

Abbildung 2.28: Services Infos Compute

Storage

Bei Storage spielt es wieder eine Rolle was für eine Art Storage es ist in diesem Beispiel ist es eine Cloud SQL Datenbank. Dabei wird oftmals anhand der Anzahl Rows oder Grösse der Datenbank abgerechnet + sollen hier auch wieder Links verfügbar sein, um auf ein Datenbankdashboard zu gelangen + die Möglichkeit geben den Service löschen zu können.

Dashboard

Services Offerings Account

WebApp Database

Info MySQL Instanz Size 10000 Rows Costs 20\$/10000 Rows	Management <button>delete</button> Links https://console.google.com/CloudSQL
--	---

Abbildung 2.29: Services Infos Storage

Network

Bei Network kann man noch einige Dinge konfigurieren von Cloud DNS bis zu Netzwerken (Subnetze etc.) auch Firewall Regeln, da bei den Cloud Anbietern oftmals nur SSH und HTTP/S zugelassener traffic ist muss man schliesslich auch die Möglichkeit haben Firewall Regeln festlegen zu können. Das könnte schlussendlich in etwa so aussehen:

Dashboard

Services Offerings Account

Firewall test Project

Info Firewall normally deny all add Rules for accepting incoming/outgoing Communication. Costs 5\$ / Month	Management <div><input type="checkbox"/> allowSSH <input type="checkbox"/> allowHTTPS</div> <div><button>create Rule</button> <button>change Rule</button> <button>delete Rule</button></div>
---	---

Abbildung 2.30: Services Infos Network

2.8.5 Accounts/Subscriptions/Projects/...

Für jeden Anbieter soll dem User eine Übersicht über die Account/Subscriptions/Projects gegeben werden, dadurch vereinfacht sich die Handhabung von mehreren Accounts und alle sind in einem Dashboard zusammengefügt (-> Security beachten). Dadurch hat man immer den Überblick für welches Projekt/Account man wie viele Services abonniert hat.

Dashboard

Services Offerings Account

Projects

Name	Services	Options
My test Project	2	+ add Service

Add project

Abbildung 2.31: Accounts Übersicht

2.9 Security

Wie bei Bitnami wäre es wohl sicherer die Zugriffsdaten für die Cloud Anbieter abzusichern (bei Bitnami wird dies über ein Vault sichergestellt), ansonsten könnte ein Angreifer ganze Instanzen bei verschiedenen Anbietern löschen oder sonstige Bösertige Absichten ausüben.

Dieser Vault soll auch durch ein zusätzliches Passwort geschützt sein und wird symmetrisch verschlüsselt (Mail Anbieter: Proton Mail¹² macht dies ebenfalls so) -> jedoch fragt Bitnami bei jedem login wieder nach dem Passwort und vergisst dann wieder alle Instanzen (ein Abgleich mit dem Anbieter wäre hier sicher nicht schlecht).

¹²Secure email: ProtonMail is free encrypted email. URL: <https://protonmail.ch>.

2.10 Fazit

Das Servicedashboard scheint soweit umsetzbar zu sein, ein Knackpunkt wird einfach noch die Absicherung der Zugriffe auf alle die Cloud Anbieter und deren Services. Z.B.: Bitnami bietet ein Download für Private Key und Putty Key File an: Und erstellt für

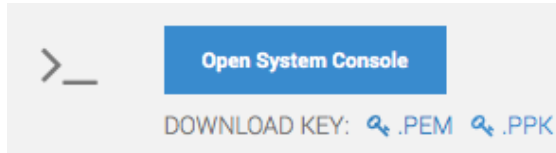


Abbildung 2.32: Bitnami Private Key

jeden Server ein neues keypaar: Was nicht gerade die Beste Lösung ist.

bitnami-launchpad-o_xif9m (65:44:c3:5e:e3:0a:8c:59:f5:1c:c3:79:bc:e9:75:2a)

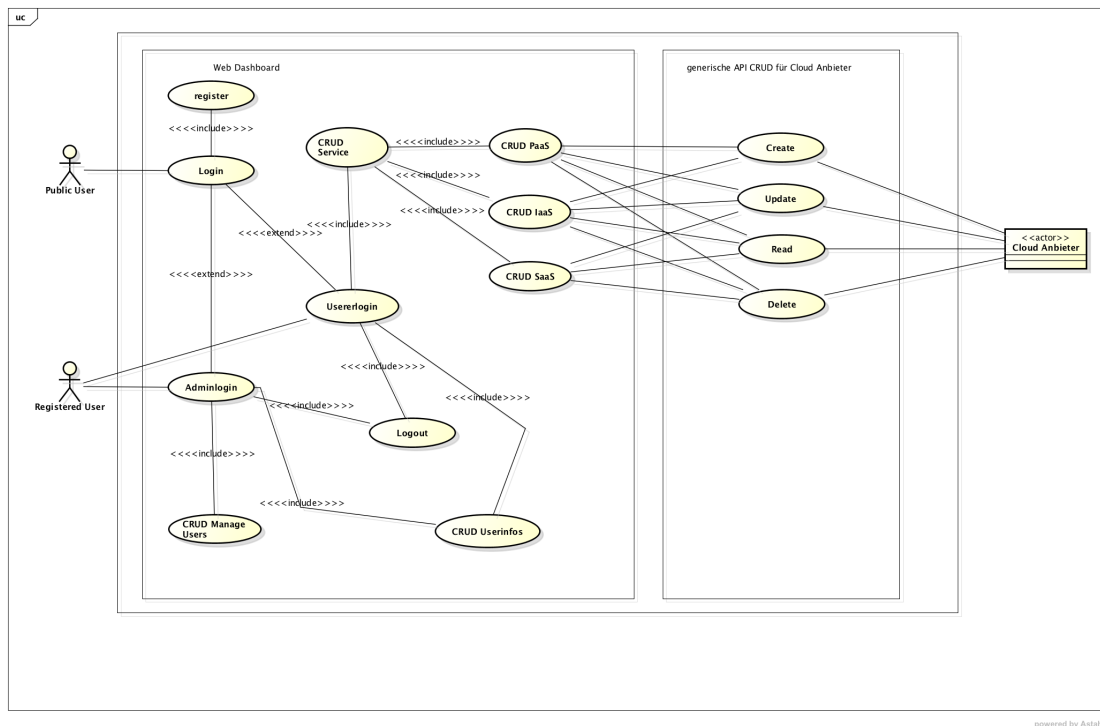
bitnami-launchpad-jfsn6oi (7c:ca:6e:18:92:ba:21:13:22:c7:29:ba:b9:0a:81:da)

Abbildung 2.33: Bitnami Key

2.11 Akteure

Akteur	Ziel
Public User	Registrieren
Registered User	Wenn User: Login Logout Service anlegen (Create) Service Infos lesen (Read) Service ändern (Update) Service löschen (Delete) Benutzerinfos ändern Wenn Admin: Benutzer anlegen Benutzer löschen Benutzer ändern Benutzerinfos ändern
Cloud Anbieter	Service anlegen (Create) Service Infos lesen (Read) Service ändern (Update) Service löschen (Delete)

2.12 Use Case Diagramm



2.13 User Stories Skizzen

2.14 Rollen

2.14.1 Public User

Public User sind alle öffentlichen Besucher des Dashboards.

2.14.2 Registered User

Der registrierte Nutzer ist Anwender des Dashboards und verwendet dieses zur Aufgabenerleichterung. Bei dem Nutzer kann es sich um einen System Administrator, DevOps, Operator oder Software Entwickler handeln, da beim Dashboard für jeden was dabei ist.

2.14.3 Admin

Der Admin ist für die Instandhaltung des Dashboards zuständig und verwaltet die User.

2.15 Ziele

Im Umfang soll die Applikation in etwa folgendes bieten:

- Registrierung (Mail Adresse/Passwort)
- Login
- Administrationoberfläche
- Benutzerinfos anpassen
- Auswahl aus mehreren Cloud Anbieter
- mehrere Cloud Accounts hinzufügen
- Abonnieren von Services (Compute/Storage/Network)
- Unterteilung der Services in Compute/Storage/Network
- Übersicht aller zur Verfügung stehenden Services
- Management der Services (erstellen/ändern/löschen)
- Links zu Loginpanels von Cloud Anbieter
- Übersicht über abonnierte Services
- Unterstützung Private Cloud (OpenStack,CloudStack, Docker(Deis -> PaaS))
- Anbieter spezifische Services anbieten
- generische API
- Anstehende Kosten anzeigen
- Einfaches hinzufügen eines Cloud Accounts (Wizard bieten)

2.16 Epic

- Service abonnieren (Compute/Storage/Network)

2.17 User Stories

2.17.1 Public User

- Als Public User möchte ich mich registrieren können
- Als Public User möchte ich mich auf Dashboard verbinden, um einloggen zu können

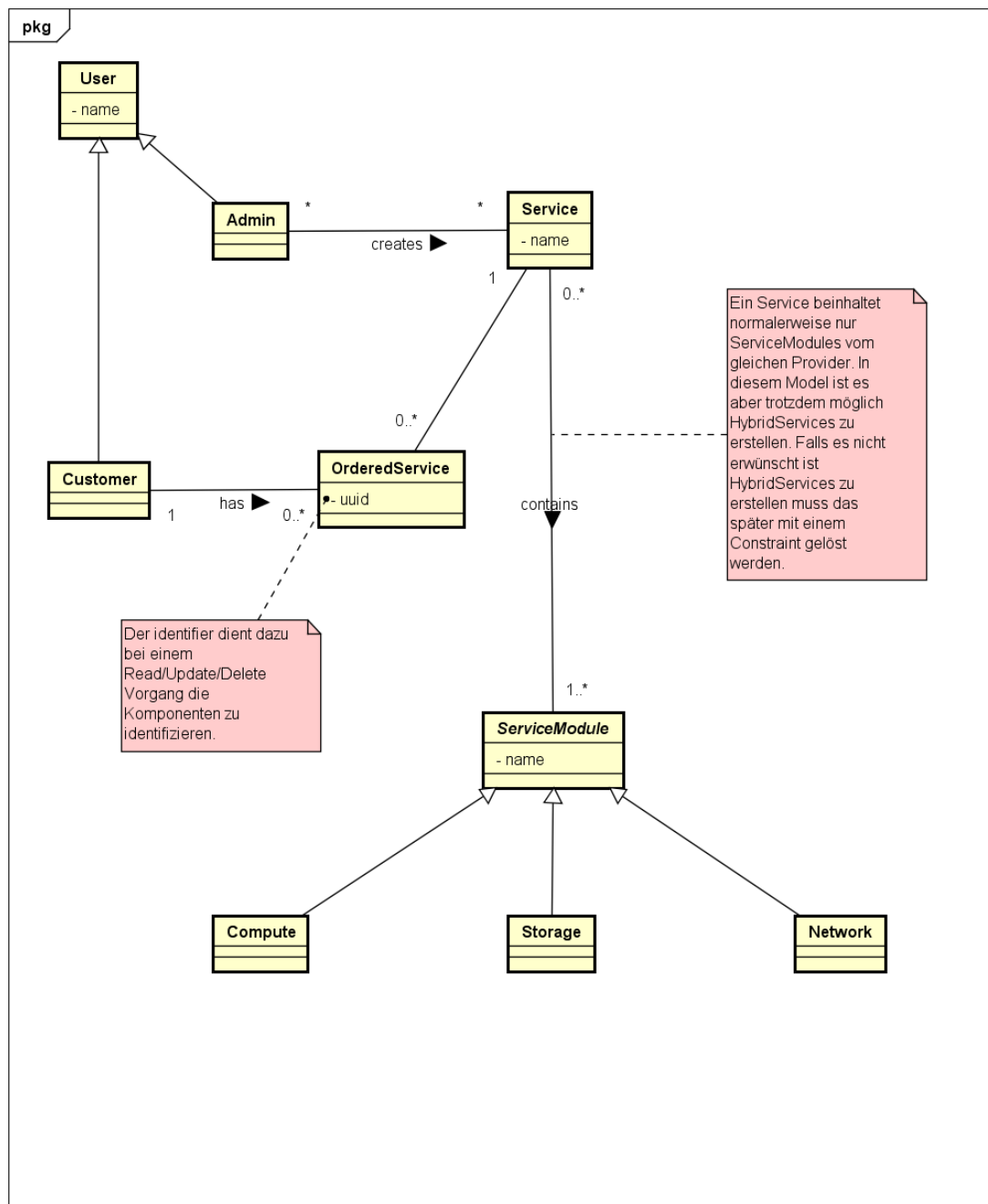
2.17.2 Registered User

- Als registered User möchte ich mich einloggen können
- Als registered User möchte ich eine Übersicht aller angebotenen Cloud Anbieter sehen
- Als registered User möchte ich Zugriff auf meine Accountinfos
- Als registered User möchte ich meine Accountinfos anpassen können
- Als registered User möchte ich ein Cloud Anbieter auswählen können, um auf die Übersicht der Offerings zu kommen
- Als registered User möchte ich eine Übersicht meiner abonnierten Services haben
- Als registered User möchte ich Services löschen können
- Als registered User möchte ich Compute Instanzen neustarten können
- Als registered User möchte ich Compute Instanzen herunterfahren können
- Als registered User möchte ich die kosten der Services angezeigt haben
- Als registered User möchte ich direkte Verlinkungen zu den Services haben

2.17.3 Admin

- Als Admin möchte ich Zugriff auf eine Administrationsoberfläche
- Als Admin möchte ich User erstellen können
- Als Admin möchte ich User löschen können
- Als Admin möchte ich User ändern können

2.18 Domainmodell Skizze



powered by Astah

3 Anforderungen

3.1 API

Die API definiert einen Workflow der einen Service auf einer Cloud erstellt. Es ist offen, ob dieser Service über mehrere Cloud Anbieter hinaus geht. Der Service wird durch ein Konfigurationsfile definiert. Die Software auf den Instanzen wird durch Images installiert. Ein Service kann auch wieder gelöscht werden. Es ist nicht die Aufgabe der API existierende Services zu identifizieren. Die API muss Modular sein, das heisst es sollte möglich sein andere oder eigene Programme für die Cloud Kommunikation zu verwenden. Innerhalb der API werden Compute, Storage, Network usw. als ServiceModule bezeichnet. Diese Abstraktion ermöglicht das wiederverwenden und erweitern der API.

3.2 Customer-Dashboard

3.2.1 Homescreen

Im Homescreen werden alle zu Verfügung stehenden Services angezeigt. Hier werden die Services Offerings genannt um eine Unterscheidung zwischen Abonnierten Services (Services) und zur Verfügung stehenden Services (Offerings) machen zu können.

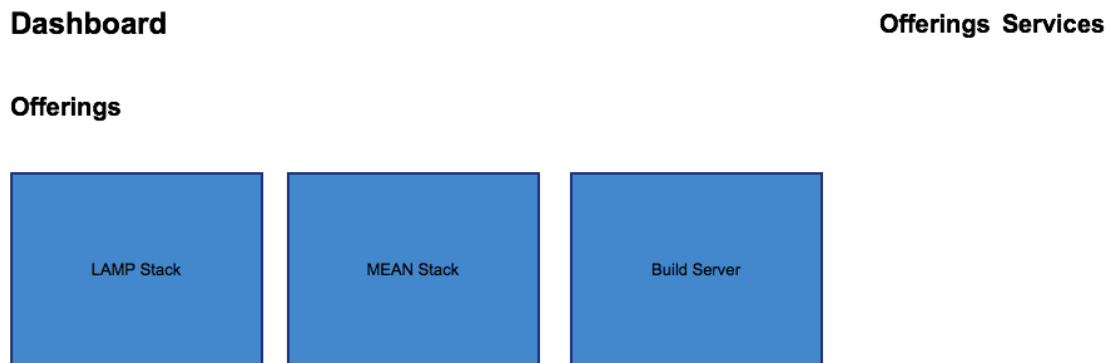


Abbildung 3.1: Homescreen Customer

3.2.2 Services Übersicht

In der Services Übersicht werden dem Customer alle abonnierten Services angezeigt und können hier auch gekündigt werden.

Dashboard

Offerings Services

Your Services

Name	Options
LAMP Stack	terminate
MEAN Stack	terminate

Abbildung 3.2: Services Übersicht

3.2.3 Service abonnieren

Sobald ein Service auf dem Homescreen ausgewählt wird und auf den "subscribe" Button geklickt wird, wird dieser abonniert und wird in der Services Übersicht angezeigt.

LAMP Stack[subscribe Service](#)

Abbildung 3.3: Services Settings

3.3 Admin-Dashboard

Zusätzlich zum Customer-Dashboard soll ein Admin-Dashboard zur Verfügung stellen in welchem der Admin Services und Servicemodule erstellen kann.

Service

Ein Service hat einen bestimmten Namen und jedem Service sind eine gewisse Anzahl Servicemodule zugeteilt. um den Service abbilden zu können. Hier kann der Admin den Service ändern und je nach Anforderung den Service anpassen.

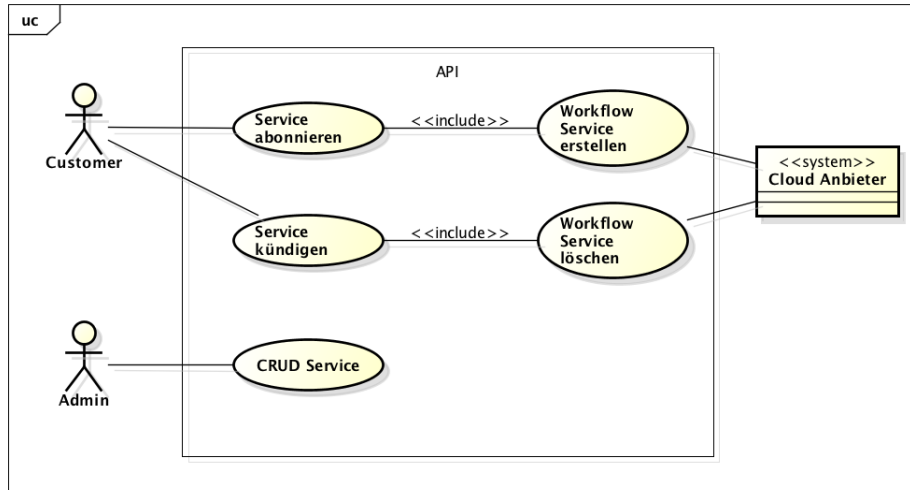
Services

Name	Aktion
LAMP Stack	delete / change
MEAN Stack	delete / change
Build Server	delete / change
create Service	

Abbildung 3.4: Homescreen Admin

3.4 Use Cases

3.4.1 Use Case Diagramm



powered by Astah

3.4.2 Aktoren & Stakeholders¹

Customer

Als Customer möchte ich meine abonnierten Services verwalten.

Aktor	Typ	Ziele
Customer	Primary	<ul style="list-style-type: none">• Service abonnieren• Service kündigen

¹Craig Larman. *UML 2 und Patterns angewendet Objektorientierter Softwareentwicklung*. mitp, 2005. ISBN: 978-3-8266-1453-8.

Admin

Als Admin möchte ich Services und Servicemodule verwalten können.

Aktor	Typ	Ziele
Admin	Primary	<ul style="list-style-type: none">• Service erstellen• Service anpassen• Service löschen• Servicemodul erstellen• Servicemodul anpassen• Servicemodul löschen

3.4.3 Beschreibungen fully dressed

UCo1: Service abonnieren

Primäraktor	Customer
Steakholders und Interessen	Customer: Möchte einen Service abonnieren
Vorbedingungen	Das Customer-Dashboard wurde geöffnet.
Nachbedingungen	Die Service Infos wurden gespeichert und der Workflow wurde angestossen
Standartablauf	<ol style="list-style-type: none">1. Wiederholen bis kein Service mehr abonniert werden soll<ol style="list-style-type: none">(a) Der Customer wechselt in die Offerings Übersicht(b) Der Customer wählt einen der vorhandenen Services aus(c) Der Customer drückt den Button subscribe Service(d) Der Customer wird in die Services Übersicht weitergeleitet2. Der Customer schliesst das Customer-Dashboard
Alternativer Ablauf	<ol style="list-style-type: none">1. (a) Der Customer entscheidet sich um<ol style="list-style-type: none">i. Schliesst das Fenster/Tab(b) Der Customer entscheidet sich um<ol style="list-style-type: none">i. Schliesst das Fenster/Tabii. geht zurück in die Offerings Übersicht(c) Der Customer entscheidet sich um<ol style="list-style-type: none">i. Schliesst das Fenster/Tabii. geht zurück in die Offerings Übersicht(d) Der Customer entscheidet sich um<ol style="list-style-type: none">i. Schliesst das Fenster/Tabii. geht zurück in die Offerings Übersicht

Spezielle Anforderungen	siehe nichtfunktionale Anforderungen
Technologie- und Datenvarianten	Keine
Auftrittshäufigkeit	mehrmals pro Woche
Offene Fragen	Keine

UCo2: Service kündigen

Primäraktor	Customer
Steakholders und Interessen	Customer: Möchte einen Service kündigen
Vorbedingungen	Das Customer-Dashboard wurde geöffnet.
Nachbedingungen	Die Service Infos wurden gelöscht und der Workflow wurde angestossen
Standartablauf	<ol style="list-style-type: none">1. Wiederholen bis kein Service mehr gekündigt werden soll<ol style="list-style-type: none">(a) Der Customer wechselt in die Services Übersicht(b) Der Customer wählt einen der vorhanden Services aus(c) Der Customer drückt auf den link terminate(d) Der Customer wird in die Services Übersicht weitergeleitet2. Der Customer schliesst das Customer-Dashboard
Alternativer Ablauf	<ol style="list-style-type: none">1. <ol style="list-style-type: none">(a) Der Customer entscheidet sich um<ol style="list-style-type: none">i. Schliesst das Fenster/Tab(b) Der Customer entscheidet sich um<ol style="list-style-type: none">i. Schliesst das Fenster/Tabii. wählt einen anderen Service(c) Der Customer entscheidet sich um<ol style="list-style-type: none">i. Schliesst das Fenster/Tabii. wählt einen anderen Service
Spezielle Anforderungen	siehe nichtfunktionale Anforderungen
Technologie- und Datenvarianten	Keine
Auftrittshäufigkeit	mehrmals pro Woche

Offene Fragen	Keine
----------------------	-------

UCo4: Service verwalten

Primäraktor	Customer
Steakholders und Interessen	Admin: Möchte einen Service verwalten
Vorbedingungen	Das Admin-Dashboard wurde geöffnet und Admin eingeloggt, falls Service gelöscht werden soll darf kein Customer mehr den Service abonniert haben
Nachbedingungen	Das Admin-Dashboard wurde geschlossen und Änderungen wurden gespeichert
Standartablauf	<ol style="list-style-type: none">1. Der Admin gibt die Webadresse für das Admin-Dashboard ein2. Wiederholen bis kein neuer Service hinzugefügt werden muss<ol style="list-style-type: none">(a) Der Admin wechselt in die Services Übersicht(b) Der Admin drückt auf den button create Service(c) Der Admin füllt die benötigten Daten ein (Name, welche Servicemodule)(d) der Admin bestätigt mit Klick auf Button Save
Alternativer Ablauf	<ol style="list-style-type: none">2. <ol style="list-style-type: none">(a) Wiederholen bis kein Service mehr geändert werden muss<ol style="list-style-type: none">i. Service auswählen und auf Link change klickenii. Daten änderniii. Durch Klick auf Button Save bestätigen(b) Wiederholen bis kein Service mehr gelöscht werden muss<ol style="list-style-type: none">i. Service auswählen und Link löschen auswählen
Spezielle Anforderungen	siehe nichtfunktionale Anforderungen

Technologie- und Datenvarianten	Keine
Auftrittshäufigkeit	mehrmals pro Woche
Offene Fragen	Keine

3.5 Epics

3.5.1 Customer

- Service abonnieren
- Service kündigen

3.5.2 Admin

- Service verwalten
- Servicemodul verwalten

3.6 User Stories

3.6.1 Rollen

Customer

Als Customer benutze ich das Dashboard, um für mich einen Service zu abonnieren oder zu kündigen, ebenfalls verwalte ich meine Cloud Login Daten

Admin

Als Admin erstelle ich neue Services und Servicemodule und erweitere diese um neue Funktionen/Verbesserungen.

3.6.2 Customer

Customer abonniert Service

Priorität	Hoch
Story Points	4
Story	Als Customer möchte ich einen Service abonnieren können
Akzeptanzkriterien	
A1	Der Customer kann einen Service abonnieren
A3	Storage,Compute und Network wurden, wie im Service beschrieben erstellt

Customer kündigt Service

Priorität	Hoch
Story Points	4
Story	Als Customer möchte ich einen Service kündigen können
Akzeptanzkriterien	
A1	Der Customer kann einen Service kündigen
A3	Storage, Compute und Network werden, wie im Service beschrieben gelöscht

Customer will abonnierte Services sehen

Priorität	Hoch
Story Points	2
Story	Als Customer möchte ich sehen welche Services ich abonniert habe.
Akzeptanzkriterien	
A1	Der Customer kriegt eine Liste mit seinen abonnierten Services zurück

Customer will verfügbare Services angezeigt bekommen

Priorität	Hoch
Story Points	2
Story	Als Customer möchte ich sehen welche Services ich abonnieren kann.
Akzeptanzkriterien	
A1	Der Customer kriegt eine Liste mit seinen zur Verfügung stehenden Services zurück

Customer geht in die Offerings Übersicht

Priorität	Hoch
Story Points	6
Story	Als Customer möchte ich die Offerings in einer Übersicht ansehen können
Akzeptanzkriterien	
A1	Der Customer kann im Dashboard in die Offerings Übersicht wechseln.

Customer geht in die Service Übersicht

Priorität	Hoch
Story Points	6
Story	Als Customer möchte ich meine abonnierten Services in einer Übersicht angezeigt bekommen
Akzeptanzkriterien	
A1	Der Customer kann seine abonnierten Services in einer Übersicht anzeigen

Customer will Informationen über abonnierte Services sehen

Priorität	Hoch
Story Points	6
Story	Als Customer möchte ich Informationen über abonnierte Service einsehen können.
Akzeptanzkriterien	
A1	Der Customer kann abonnierten Service auswählen und kriegt Infos zu den Servicemodulen

3.6.3 Admin

Admin erstellt Service

Priorität	Hoch
Story Points	6
Story	Als Admin möchte ich Services erstellen können
Akzeptanzkriterien	
A1	Der Service kann nur erstellt werden, falls Admin eingeloggt ist.
A2	Als Admin krieg ich die Übersicht der verfügbaren Services
A3	Dem Service können Servicemodule hinzugefügt werden
A4	Service kann erstellt werden
A5	Der Service ist erstellt

Admin ändert Service

Priorität	Hoch
Story Points	6
Story	Als Admin möchte ich Services ändern können
Akzeptanzkriterien	
A2	Als Admin krieg ich die Übersicht der verfügbaren Services
A3	Dem Service können Servicemodule hinzugefügt werden
A4	Service kann geändert werden
A5	Der Service ist geändert

Admin löscht Service

Priorität	Hoch
Story Points	6
Story	Als Admin möchte ich Services löschen können
Akzeptanzkriterien	
A5	Der Service ist gelöscht

Admin greift auf Dashboard zu

Priorität	Hoch
Story Points	1
Story	Als Admin möchte ich auf das Admin-Dashboard zugreifen können
Akzeptanzkriterien	
A1	Der Admin kann den Url des Customer-Dashboard aufrufen und kriegt ein Login angezeigt.

Admin geht in die Service Übersicht

Priorität	Hoch
Story Points	2
Story	Als Admin möchte ich einen Überblick über die vorhanden Services
Akzeptanzkriterien	
A1	Der Admin kann die Service Übersicht öffnen

Admin Konfigurationsdatei im Servicemodul hinterlegen

Priorität	Hoch
Story Points	2
Story	Als Admin möchte ich dem Servicemodul eine Konfigurationsdatei hinterlegen
Akzeptanzkriterien	
A1	Der Admin kann dem Servicemodul eine Konfigurationsdatei hinterlegen

Admin erstellt Servicemodul

Priorität	Hoch
Story Points	4
Story	Als Admin möchte ich Servicemodule erstellen können
Akzeptanzkriterien	
A2	Das Servicemodul wird erstellt und wird in der Servicemodule Übersicht angezeigt

Admin ändert Servicemodul

Priorität	Hoch
Story Points	4
Story	Als Admin möchte ich Servicemodule ändern können
Akzeptanzkriterien	
A1	Als Admin krieg ich die Übersicht der verfügbaren Servicemodule
A2	Als Admin kann ich das Servicemodule anpassen und speichern
A3	Änderungen werden gespeichert

Admin löscht Servicemodul

Priorität	Hoch
Story Points	4
Story	Als Admin möchte ich Servicemodule löschen können
Akzeptanzkriterien	
A1	Als Admin krieg ich die Übersicht der verfügbaren Servicemodule
A2	Servicemodule ist gelöscht

Admin geht in die Servicemodul Übersicht

Priorität	Hoch
Story Points	2
Story	Als Admin möchte ich einen Überblick über die vorhanden Servicemodule
Akzeptanzkriterien	
A1	Der Admin kann die Servicemodule Übersicht öffnen
A2	Die Servicemodule Übersicht wird angezeigt.

3.7 Nicht-funktionale Anforderungen

3.7.1 Menge

- Die Software unterstützt mindestens 1 Storage Anbieter
- Die Software unterstützt mindestens 1 Compute Anbieter
- Die Software unterstützt mindestens 1 Network Anbieter
- Es soll für Compute, Storage, Network mindestens je 1 Servicemodul erstellt werden

3.7.2 Schnittstellen

- Die Software wird über HTTP/HTTPS angesprochen
- Zur Interaktion im Admin-Dashboard/Customer-Dashboard werden die herkömmlichen Schnittstellen gebraucht (Maus,Tastatur,Bildschirm)

3.7.3 Qualitätsmerkmale

Funktionalität

siehe Abschnitt API und Dashboard

Zuverlässigkeit

- Der Workflow zum erstellen eines Services soll entweder durchgeführt und abgeschlossen werden oder falls Unterbruch/Fehler rückgängig gemacht werden.
- Die Software soll verteilt betrieben werden und eine möglichst hohe Verfügbarkeit/Zuverlässigkeit bieten

Benutzerbarkeit

- Konfigurationen können über das vorgesehene Admin-Dashboard geändert werden
- Zum verwenden der Software besteht noch ein einfaches User-Dashboard

Effizienz

- Die Software Soll mehrere Aufträge von Customern gleichzeitig abarbeiten können

Änderbarkeit

Die Software soll modular aufgebaut werden, damit Erweiterungen in Zukunft problemlos möglich sind.

Übertragbarkeit

Das Projekt wird in Java geschrieben und ist somit also auf Java mindestens in der Version 1.8 angewiesen.

4 Design

4.1 REST API

4.2 CRUD Service

4.2.1 Services

URI Path /services
Methods GET

4.2.2 JSON Response

```
{  
  id : <number>,  
  serviceName : <string>,  
  serviceModules:[ category:<string>, size:<string>, name:<string> },  
  ...] },  
  ...  
}
```

4.2.3 Service

URI Path /services/{id}
Methods GET, PUT, DELETE, POST

4.2.4 JSON Response

```
{  
  id : <number>,  
  serviceName : <string>,  
  serviceModules:[ category:<string>, size:<string>, name:<string> },  
  ...] }
```


4.3 CRUD OrderedService

4.3.1 OrderedServices

URI Path /OrderedServices
Methods GET

4.3.2 JSON Response

```
{  
  id : <number>,  
  orderedServiceName : <string>,  
  identifiers:[ id:<number>, uuid:<string>, category:<string>, size:<string> },  
  ...] },  
  ...  
}
```

4.3.3 OrderedService

URI Path /OrderedServices/{id}
Methods GET, DELETE

4.3.4 JSON Response

```
{  
  id : <number>,  
  orderedServiceName : <string>,  
  identifiers:[ id:<number>, uuid:<string>, category:<string>, size:<string> },  
  ...] }
```

4.4 Generic API

4.5 Operationen

4.5.1 Allgemein

- connect(uri: String, ReadOnly: Boolean)
- disconnect()

Contracts

Operation	connect(uri : String, ReadOnly: Boolean)
Cross References	UCo1
Preconditions	Eine URI wurde mitgegeben und ReadOnly True oder False gesetzt
Postconditions	<ul style="list-style-type: none">• Client ist mit dem Hypervisor verbunden
Operation	disconnect()
Cross References	UCo1
Preconditions	Client ist mit Hypervisor verbunden
Postconditions	<ul style="list-style-type: none">• Verbindung zum Hypervisor wurde geschlossen

4.5.2 Compute

- createCompute(Config: String)
- deleteCompute(comp: Compute)
- getCompute(name: String)

Contracts

Operation	createCompute(Config: String)
Cross References	UCo1
Preconditions	Client ist mit Hypervisor verbunden
Postconditions	<ul style="list-style-type: none">• Compute Instanz wurde erstellt
Operation	deleteCompute(comp: Compute)
Cross References	UCo1
Preconditions	Client ist mit Hypervisor verbunden
Postconditions	<ul style="list-style-type: none">• Compute Instanz wurde gelöscht
Operation	getCompute(name: String)
Cross References	UCo1
Preconditions	Client ist mit Hypervisor verbunden
Postconditions	<ul style="list-style-type: none">• Hypervisor gibt ComputeObjekt zurück

4.5.3 Storage

- createStorage(Config: String)
- deleteStorage(storage: Storage)
- getStorage(name: String)

Contracts

Operation	createStorage(Config: String)
Cross References	UCo1
Preconditions	Client ist mit Hypervisor verbunden
Postconditions	<ul style="list-style-type: none">• Storage wurde erstellt
Operation	deleteStorage(storage: Storage)
Cross References	UCo2
Preconditions	Client ist mit Hypervisor verbunden
Postconditions	<ul style="list-style-type: none">• Storage wurde gelöscht
Operation	getStorage(name: String)
Cross References	UCo2
Preconditions	Client ist mit Hypervisor verbunden
Postconditions	<ul style="list-style-type: none">• Hypervisor gibt Storage Objekt zurück

4.5.4 Network

- createNetwork(Config: String)
- deleteNetwork(network: Network)
- getNetwork(name: String)

Contracts

Operation	createNetwork(Config: String)
Cross References	UCo1
Preconditions	Client ist mit Hypervisor verbunden
Postconditions	<ul style="list-style-type: none">• Netzwerk wurde erstellt
Operation	deleteNetwork(network: Network)
Cross References	UCo2
Preconditions	Client ist mit Hypervisor verbunden
Postconditions	<ul style="list-style-type: none">• Netzwerk wurde gelöscht
Operation	getNetwork(name: String)
Cross References	UCo2
Preconditions	Client ist mit Hypervisor verbunden
Postconditions	<ul style="list-style-type: none">• Hypervisor gibt Network Objekt zurück

4.6 Configfile

Das Configfile beinhaltet die Config für das jeweilige Servicemodul (Compute,Storage,Network), dabei gibt es für Storage und Compute 3 Grössen (S,M,L), dazu wird dann das nötige Configfile ausgewählt womit der Service konfiguriert und aufgesetzt wird. Mit Ausnahme von Network, wo es keine Grössen gibt und je nach vorgehensweise durch einen "Typ" (Bridge,Subnetz etc.) oder direkt durch Config hinterlegen gelöst wird.

4.6.1 Allgemein

Da je nach Bedarf Compute, Storage oder Netzwerk zuerst erstellt werden muss wird dies über eine Art Workflow behandelt, welcher entscheidet welches Servicemodul zuerst erstellt werden soll.

4.6.2 Compute

In Compute werden Memory, vCPU, Harddisk und Netzwerk (IP) zugewiesen. Es gibt noch einige mehr Funktionen, diese werden zu diesem Zeitpunkt aber noch nicht

behandelt.

Name/uuid

Jeder Compute Instanz muss ein eindeutiger Name gegeben werden, dieser wird zu beginn direkt in das Configfile eingefügt und zu einem späteren Zeitpunkt soll es möglich sein den Namen via Customer-Dashboard festzulegen. Bei libvirt wird ebenfalls eine uuid benötigt, welche eine eindeutige Identifizierung erlaubt -> wodurch das auffinden und löschen einer Instanz erleichtert werden soll

Memory

Memory wird entweder als fester Wert mitgegeben oder wird über die Instanzgrösse beim Cloud Anbieter vorbestimmt (micro,medium,large).

vCPU

vCPU's werden entweder als Wert mitgegeben oder durch den Cloud Anbieter vorgegeben (die Instanzgrösse).

Image

Je nach Anforderung muss der Pfad zu einer Boot ISO mitgegeben werden (für die Installation eines Betriebssystems)

Boot Disk

Ebenfalls muss noch eine Boot Disk übergeben werden, dies kann ein bereits bestehender Storage Pool,Volume oder eine Datei sein. Die Grösse wird hier durch die ausgewählte Disk/Datei bestimmt, bei Cloud Anbietern jedoch durch die Instanzgrösse.

Network

Beim Netzwerk kann eine IP oder MAC Adresse mitgegeben oder automatisch zugewiesen werden.

4.6.3 Storage

Name

Storage besitzt einen festen eindeutigen Namen über welchen der Pool angesprochen werden kann.

Grösse

Der Storage benötigt eine gewisse Grösse um erstellt zu werden oder falls es sich um eine Partition oder Datei handelt wird sie dadurch vorgegeben.

Typ

Der Storage kann sowohl lokaler als auch Netzwerk Speicher sein, hier wird zwischen verschiedenen Typen unterschieden. Z.B.: können NFS Storages eingebunden werden oder GlusterFS bzw. Sheepdog.

4.6.4 Network

Name

Network besitzt einen eindeutigen Namen, welcher nur einmal auf dem System vorhanden sein darf.

IP Family

Je Nach Konfiguration muss noch angegeben werden ob IPv4 oder IPv6 Adressen verwendet werden sollen.

IP Range

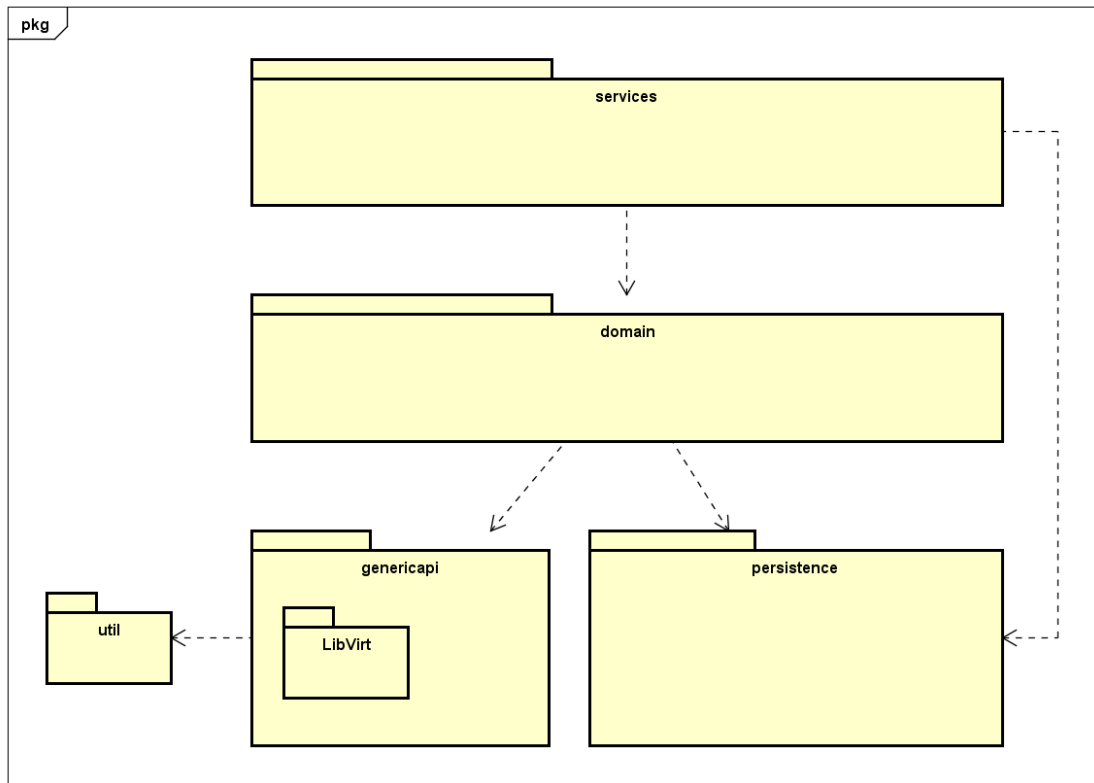
Bei der Konfiguration Bspw.: einer Bridge kann auch noch ein IP Range mitgegeben werden, welcher an die Angeschlossenen Compute Instanzen an der Bridge verteilt werden sollen.

4.7 Architektur

4.8 Systemübersicht

4.9 Logische Architektur

- services: Web-Schnittstelle (RESTful)
- genericapi: Abstraktion für Compute, Storage und Network
- util: Tools zur Bearbeitung von Configfiles
- presistence: OR-Mapping

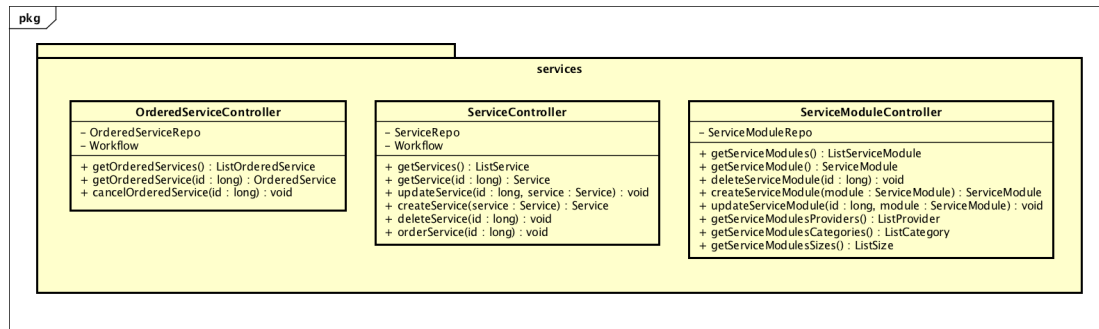


powered by Astah

4.10 Klassenstruktur

4.10.1 services

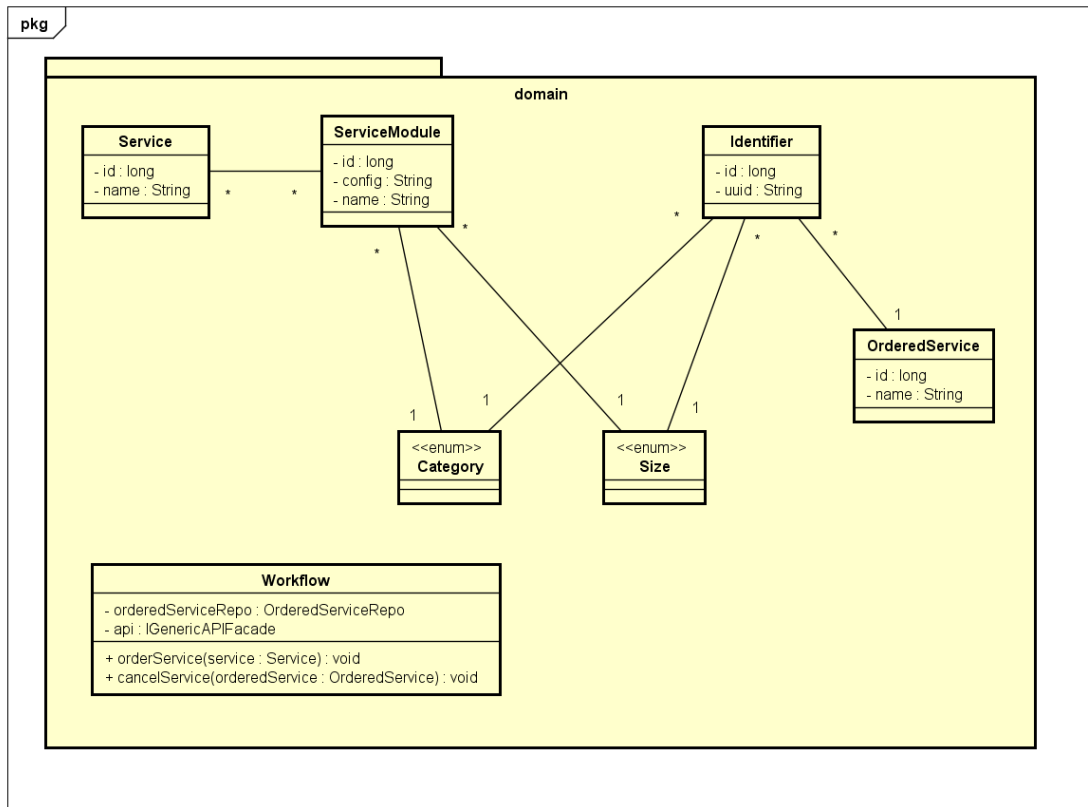
In der Schicht services wird die RestAPI implementiert.



powered by Astah

4.10.2 domain

Die Domain beinhaltet die Entitäten, die mittels OR-Mapper mit der Datenbank synchronisiert werden. Der Workflow erstellt/löscht einen Service.



4.10.3 GenericAPI

ServiceModuleHandler

Der ServiceModuleHandler entscheidet an welchen ResourceController ein Service-Module/Identifier übergeben wird. Der Entscheidungsalgorithmus kann auf verschiedenste Weise implementiert werden, deshalb gibt es ein Interface.

ResourceController

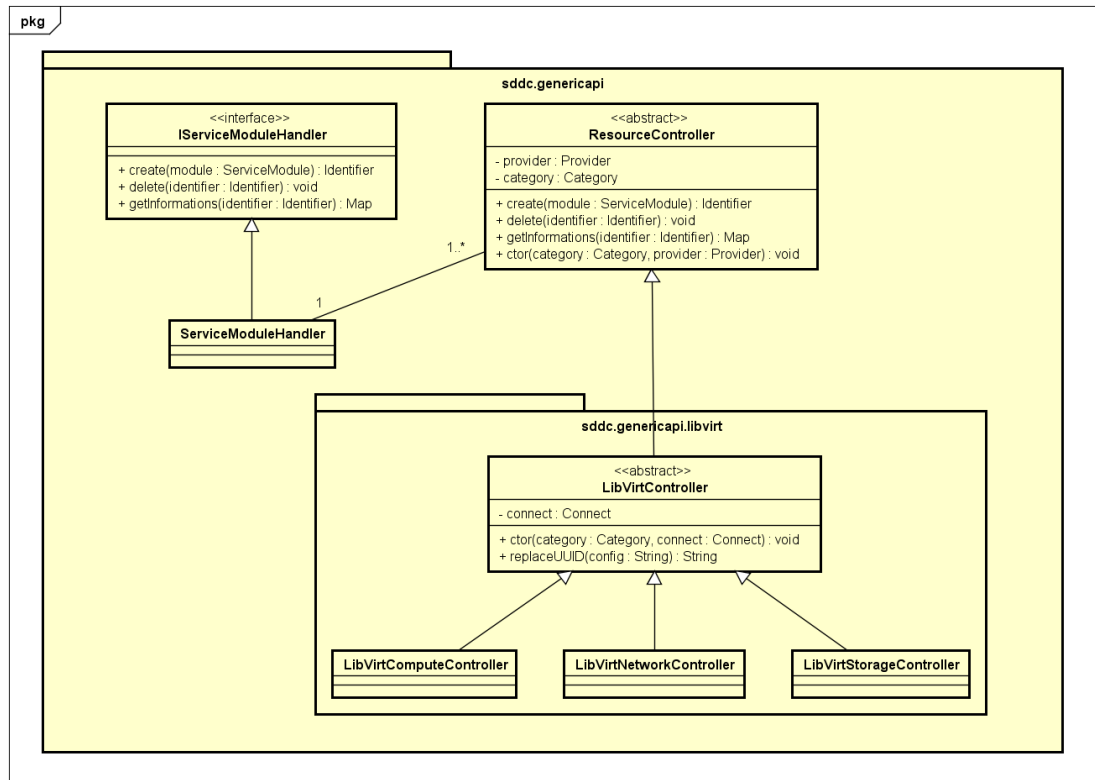
Der ResourceController ist eine Abstraktion eines Controllers, die nicht auf irgendeine Bibliotheksabhängigkeiten (z.B. LibVirt, JClouds) eingeht und lediglich dem ServiceModuleHandler die nötigen Informationen anbietet.

LibVirtController

Der LibVirtController ist eine weitere Abstraktion eines Controllers, die aber speziell einer Bibliothek (in diesem Fall LibVirt) zugewiesen ist. Die Klasse kümmert sich um die Instanziierung und kann, wenn nötig, Helfermethoden anbieten.

LibVirt<category>Controller

Die konkrete Klasse des Controllers implementiert nur noch die abstrakten Methoden des ResourceControllers.



powered by Astah

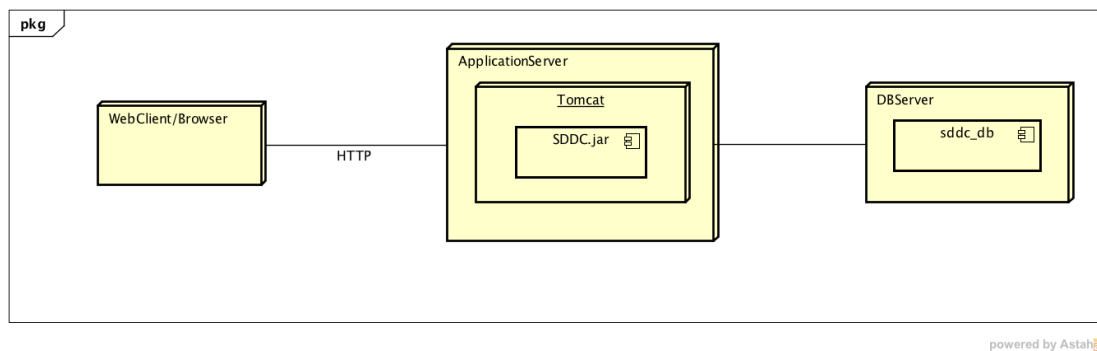
4.10.4 persistence

Wird durch OR Mapper (Hibernate) vorgenommen, dieser mappt die Java Objekte auf die Datenbank.

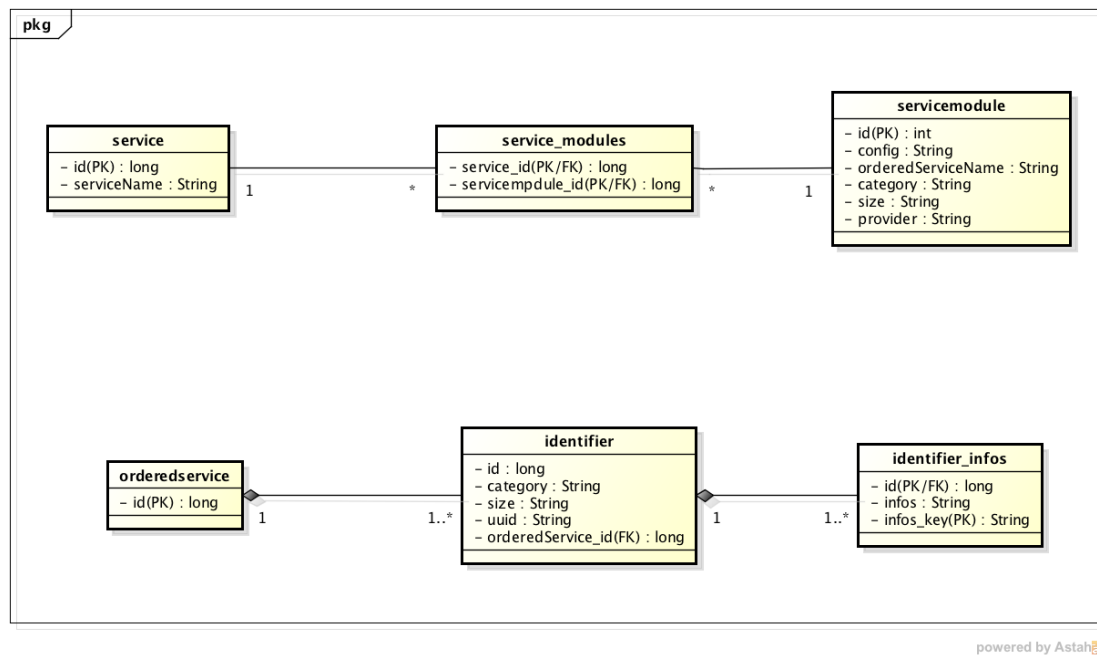
4.11 Dependency Injection

Die GenericAPI ist relativ komplex ausgefallen und kann erweitert werden. Um unnötigen Code in den oberen Schichten zu vermeiden, wird die GenericAPI mit Hilfe von Dependency Injection aufgesetzt. Es ist möglich in einem XML die gesamte GenericAPI nach belieben zusammenzusetzen.

4.12 Deployment

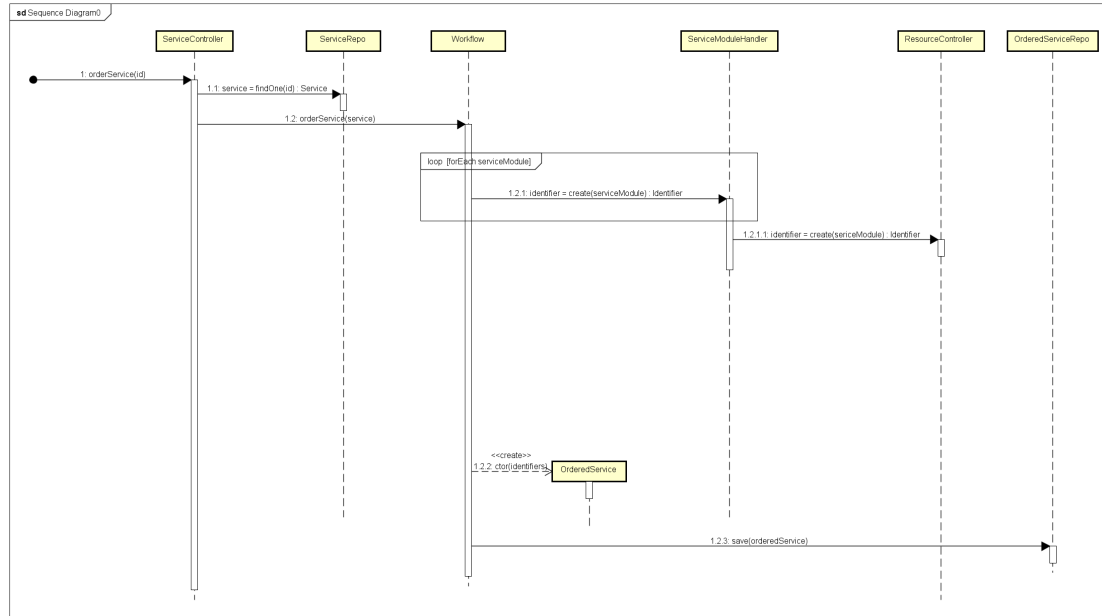


4.13 Persistierung



4.14 Sequenzdiagramme

4.14.1 service bestellen



5 Appendix

6 Glossar

Begriff	Beschreibung
Slack	Slack ist eine Software, die Kommunikation im Team und Notifications von Commits oder Builds erlauben
libcloud	Eine Library, die in Python geschrieben ist und ein generisches Interface zur Verfügung stellt um mehr als 30 Cloud Anbieter anzusprechen.
Bitnami	Ein Dashboard Anbieter um vorkonfigurierte Applikationen schnell in die Cloud stellen zu können.
libvirt	Eine Library, die viele bekannte Hypervisor unterstützt und deren Konfiguration zulässt
jclouds	Eine Library, die viele bekannte Public Cloud und Private Cloud Anbieter unterstützt

7 Abkürzungsverzeichnis

SDDC Software Defined Data Center