April, 26th 2019

Alexander Stock

Cloud Infrastructure Architect

Billing with Cloudstack

About Me

- Cloud Infrastructure Architect @itelligence

- Experience in Vmware, KVM, Nagios and Ansible

- Working with CloudStack since 2015

- Mail:

  alexander.stock@itelligence.de

CloudStack Berlin & Dresden, Germany
https://www.meetup.com/german-CloudStack-user-group
Ansible Dresden, Germany
https://www.meetup.com/Ansible-Dresden
Ceph Dresden, Germany
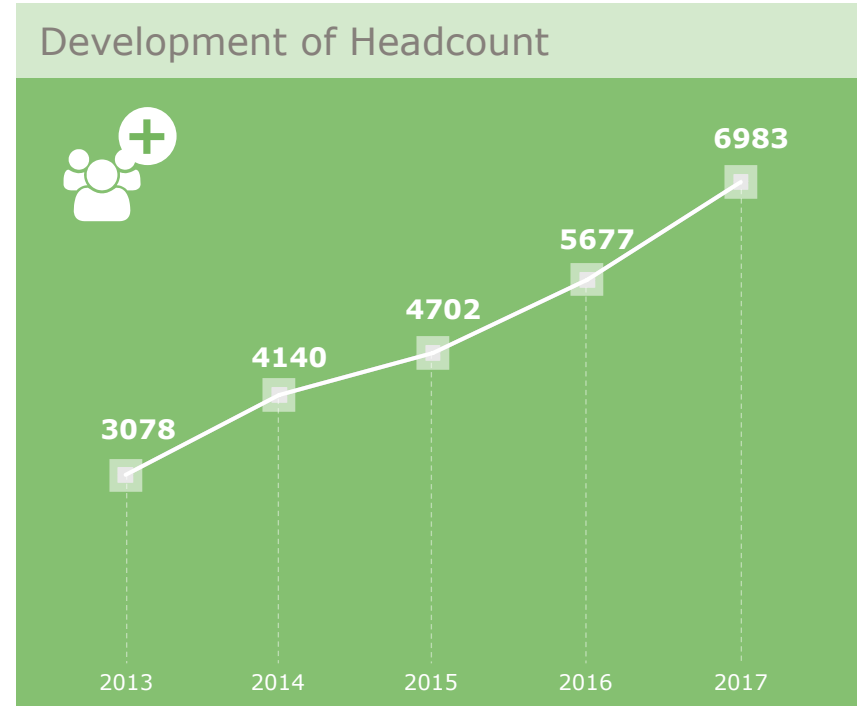https://www.meetup.com/Ceph-Dresden/

itelligence Worldwide in Numbers

Founded
1989

Employees
7,300+
(as of June 2018)

Development of Headcount
3078 (2013)
4140 (2014)
4702 (2015)
5677 (2016)
6983 (2017)

Revenue
872.2 MEUR
(2017)

Ø17,5% Revenue Growth
CAGR over the past 5 years

457 (2013)
557 (2014)
696 (2015)
778 (2016)
872 (2017)

Revenue per Region (2017)
- 48% German-Speaking Europe
- 23% Western Europe
- 10% Eastern Europe
- 17% USA
- 2% Asia
- 1% Other

Customers
6,000+

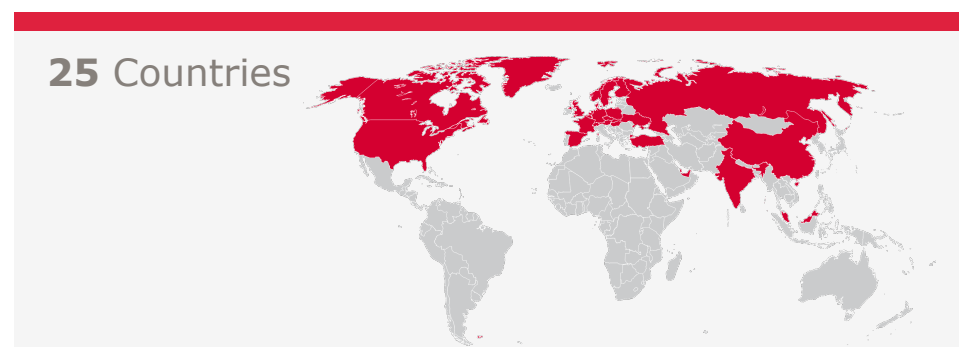## 25 Countries
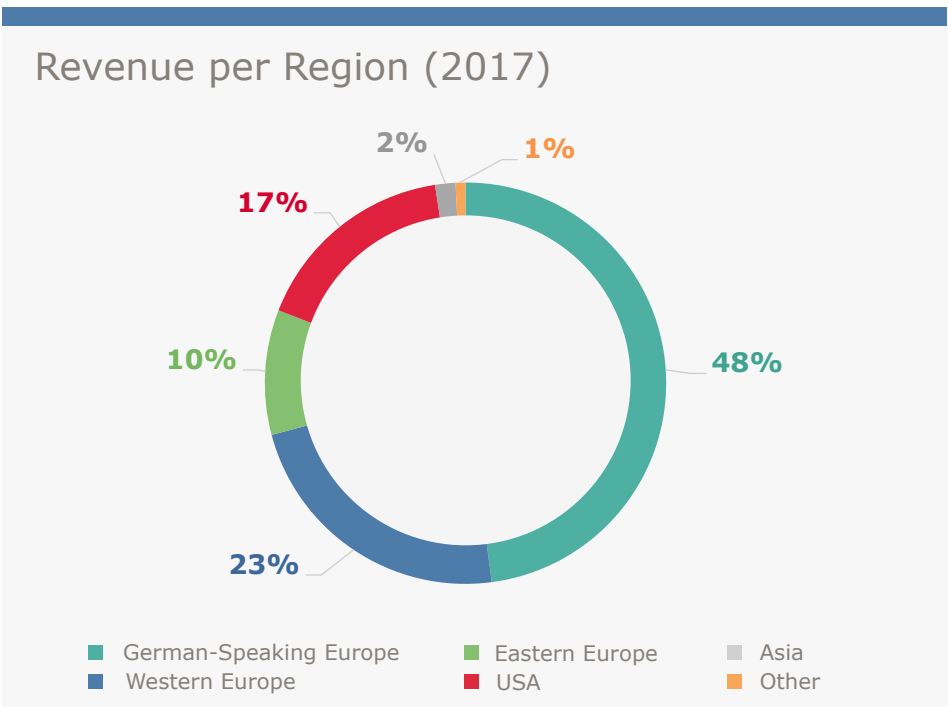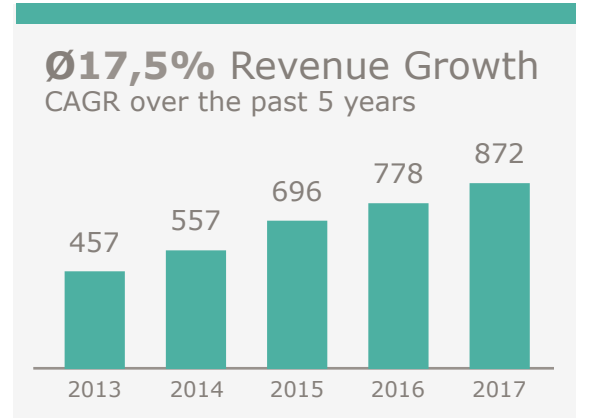
Agenda

1. Short introduction to Cloudstack billing functions
2. Our infrastructure and customer design
3. Our first steps with Cloudstack billing
4. New approach as a webservice
5. Outlook

Short introduction to Cloudstack billing functions

Short introduction to Cloudstack billing

- What information does Cloudstack provide
  - The following metrics can be queries for a specific time range:
    - CPU and memory usage of a VM
    - disk size of a volume and snapshots
    - Network usage of a virtual router
    - IP address usage
    - Template, ISO usage
    - Loadbalancer/VPN usage

Short introduction to Cloudstack billing

- ## The usage service:
  - Runs periodic jobs to generate usage records
  - Standard period: 24 hours
  - Standard settings: VM/Volume with runtime < 24 hours will not be tracked
  - Can be changed in global Cloudstack settings:

Home > Global Settings >

Select view: Global Setting ⌄   usage.stats 🔍

Name	Description	Value	Actions
usage.stats.job.aggregation.range	The range of time for aggregating the user statistics specified in minutes (e.g. 1440 for daily, 60 for hourly.	60	✎
usage.stats.job.exec.time	The time at which the usage statistics aggregation job will run as an HH24:MM time, e.g. 00:30 to run at 12:30am.	00:15	✎

Short introduction to Cloudstack billing

- **How does Cloudstack generate usage records**

  - Step1:
    - Every event like „create", „destroy", „start" or „stop" will be written to cloud.usage_event table

  - Step2:
    - Usageserver: copy new events to various table in cloud_usage database (helpertables)
    - Aggregate all data in cloud_usage.cloud_usage
    - Records can now be queried over the API

Short introduction to Cloudstack billing

- How does Cloudstack provide these information over API
  - **listUsageTypes (get mapping for usagetypes)**
    - Output:
      - usagetypeid
      - description
  - **listUsageRecords (get records):**
    - Input:
      - startdate (Date in Format: yyyy-MM-dd HH:mm:ss)
      - enddate (Date in Format: yyyy-MM-dd HH:mm:ss)
      - type (Integer for the specific usage: VM, Volume…)
      - domaindid
      - projectid
      - usageid

Short introduction to Cloudstack billing

- **How the information is structured:**
  - Sample for type 1 (Running VM):

```
name = testhost
cpunumber = 1
cpuspeed = 2000
description = testhost running time (ServiceOffering: 18) (Template: 242)
domain = itelligence
domainid = <domain-id>
enddate = 2018-09-19'T'03:59:59+00:00
memory = 1024
offeringid = <offering-id>
project = play
projectid = <project-id>
rawusage = 1
startdate = 2018-09-19'T'03:00:00+00:00
tags:
templateid = <template-id>
type = KVM
usage = 1 Hrs
usageid = <usage-id>
usagetype = 1
virtualmachineid = <vm-id>
zoneid = <zone-id>
```

Short introduction to Cloudstack billing

- Alternative ways to get usage data:

  - Direct SQL access:

    - PRO: Access to deleted objects possible

    - CON: security risk, more complicated than API calls

  - Capture Cloudstack events over API or RabbitMQ

    - PRO: saves syncronisation resources through eventbased handling

    - CON: complex setup needed (RabbitMQ,…)

Our infrastructure and customer design

Our infrastructure and customer design

- **Our Setup:**
  - We offer: automated Application/SAP setups
  - Cloudstack 4.11.2
  - Advanced Networking
  - KVM as Hypervisor
  - Ceph as Storage Backend
  - Check_MK as Monitoring Solution
  - Ansible is responsible for deploying and configuring our VMs
  - We use projects the separate resources for the customer
  - We have a self written user portal which manages Cloudstack and ansible
  - Customers doesn´t have Cloudstack access

Our infrastructure and customer design

- ## Structure of resources

  - Cloudstack Domain:

    - respresents the customer, like the internal customer „itelligence"

  - Cloudstack Project:

    - projects are used to separate different customer landscapes inside domains

    - infrastructure Services instance in each project/landscape (automation,mon,bkp)

Our first steps with Cloudstack billing

Our first steps with Cloudstack billing

- Quick solution to get an overview of the consumed resources

- Former team member wrote bash scripts in a very short amount of time

- Queried data from Cloudmonkey (API) and directly from the MySQL Database

- Small part of Jasper Reports used to generated documents in different formats

- Files were sent via mail to the administrators

Admin-Server with
billing bash scripts

API-Access for usage records and current objects

OnDemand Report generation

DB Access for deleted objects
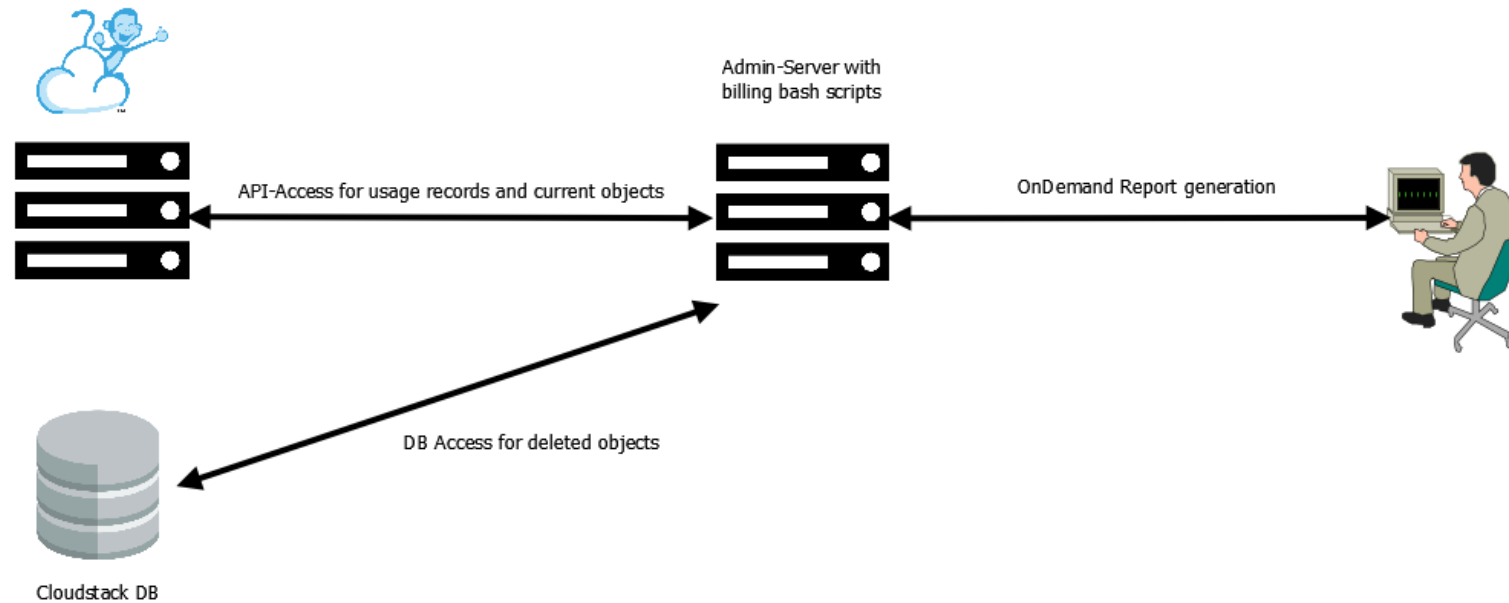
Cloudstack DB

Our first steps with Cloudstack billing

- Problems:
  - High amount of bashscripts which depended on each other were hard to maintain
  - No unique datasource (API + some information from MySQL)
  - No own database for prices/discounts and other information
  - No UI/API
  - All information had to be queried from Cloudstack each time

- Solution attempt:
  - Own database
  - Sync with Cloudstack
  - Build UI/API

New approach as a webservice

New approach as a webservice

- Features of the billing system:
    - Creation of different reports which can also work with filters:
        - Start date and end date
        - Domainid, Projectid, Virtualmachineid, Volumeid
    - Export reports in different formats
    - Sync of usage and metadata from Cloudstack → implemented as cronjob
    - Managing prices for resources (CPU,RAM,DISK) services (Backup, Monitoring) and packages (Gold-Package, Silver-Package)
    - Managing the allocation of services to packages (Tags on VM objects in Cloudstack)
    - Managing Discounts on Domain, Project, VM and Volume level
    - Webui and API with LDAP and local User authentication
    - Permissions based on API endpoints which can be managed via the UI

New approach as a webservice

- Synchronisation of Tags:

  - All tags of a VM or volume will be synced

  - We define special tags to represent service levels or packages

  - Billing can be deactivated through tags like „Billing:noCPU", „Billing:noRAM" or „Billing:noDisk"

  - Creation and removed dates will be considered in the reports

  - Consider moving tagging directly to billing-tool for next version

New approach as a webservice

- Components used:
  - database: MariaDB
  - backend framework: Flask (Python)
  - frontend framework: AngularJS + Bootstrap

New approach as a webservice

- The big picture:
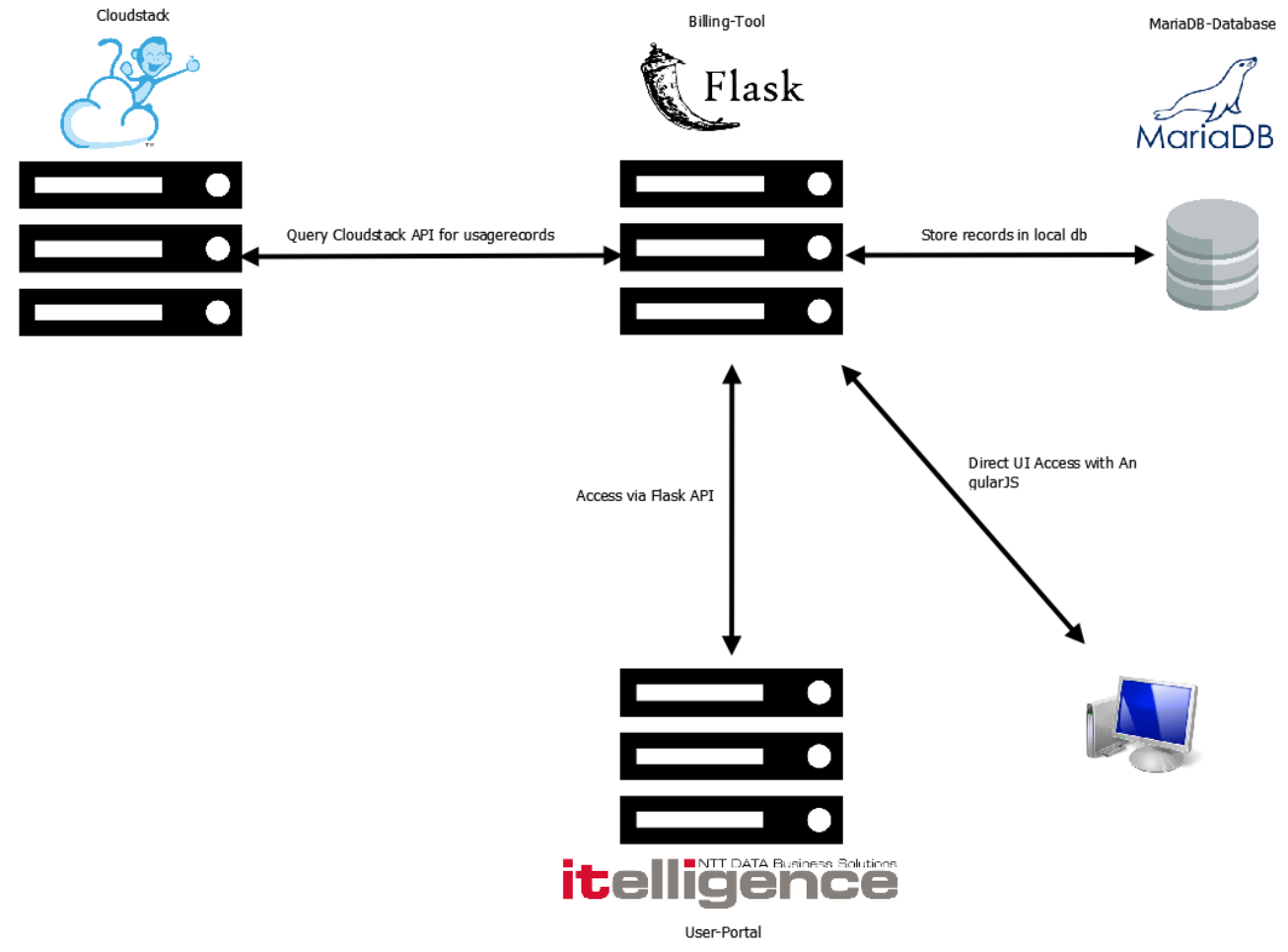
New approach as a webservice

- The sync process:
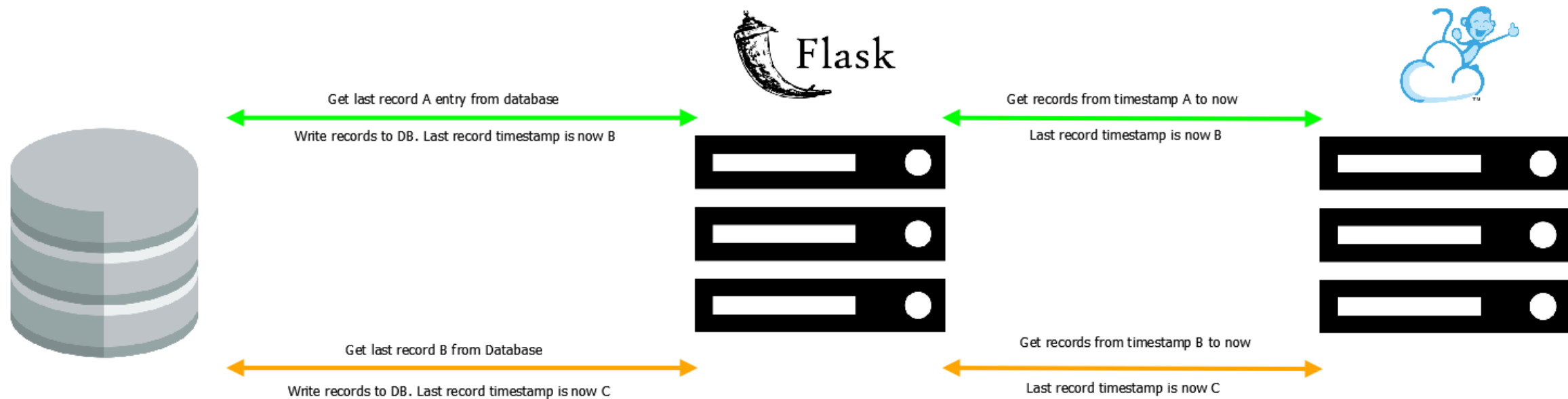
New approach as a webservice

- Functions of Report-Module:
  - Selection of different reporttypes and export to xls, csv (PDF planned)
  - Filtering: start, end, domain, project, vm, costcenter

New approach as a webservice

- Functions of Price-Module:
  - Manage prices for resources, services, packages, OS Images

New approach as a webservice

- Functions Package-Matrix-Module:
  - Add or remove services from packages

Package Matrix		
	Billing:Package-Basic	Billing:Package-Comfort
Billing:APP-Hana	☐	☐
Billing:APP-S4	☐	☐
Billing:APP-SAPRouter	☐	☐
Billing:APP-SAPWebdispatcher	☐	☐
Billing:Service-Backup	☐	☑
Billing:Service-Monitoring	☐	☑

New approach as a webservice

- Functions of Discount-Module:
  - Manage discounts for domains, projects, VMs and volumes
  - Choose duration for discounts (start, end)

New approach as a webservice

- Functions of Role-Module:
  - Management rights of roles for accessing the system
  - Rights management for different categories (create,read,update,delete)
  - Roles can be attached to users at a later stage

New approach as a webservice

- Functions of User-Module:
  - Manage local users (MariaDB)
  - Also planned for LDAP users

Outlook

- Add new export formats like PDF

- Improve UI

- Enable automatic mail generation for the customer

- Enable hard linked prices for special customers

Questions?

Contact

Alexander Stock
Cloud Infrastructure Architect
alexander.stock@itelligence.de

itelligence Global Managed Services GmbH

We make the most of SAP® solutions!

Copyright itelligence AG - All rights reserved