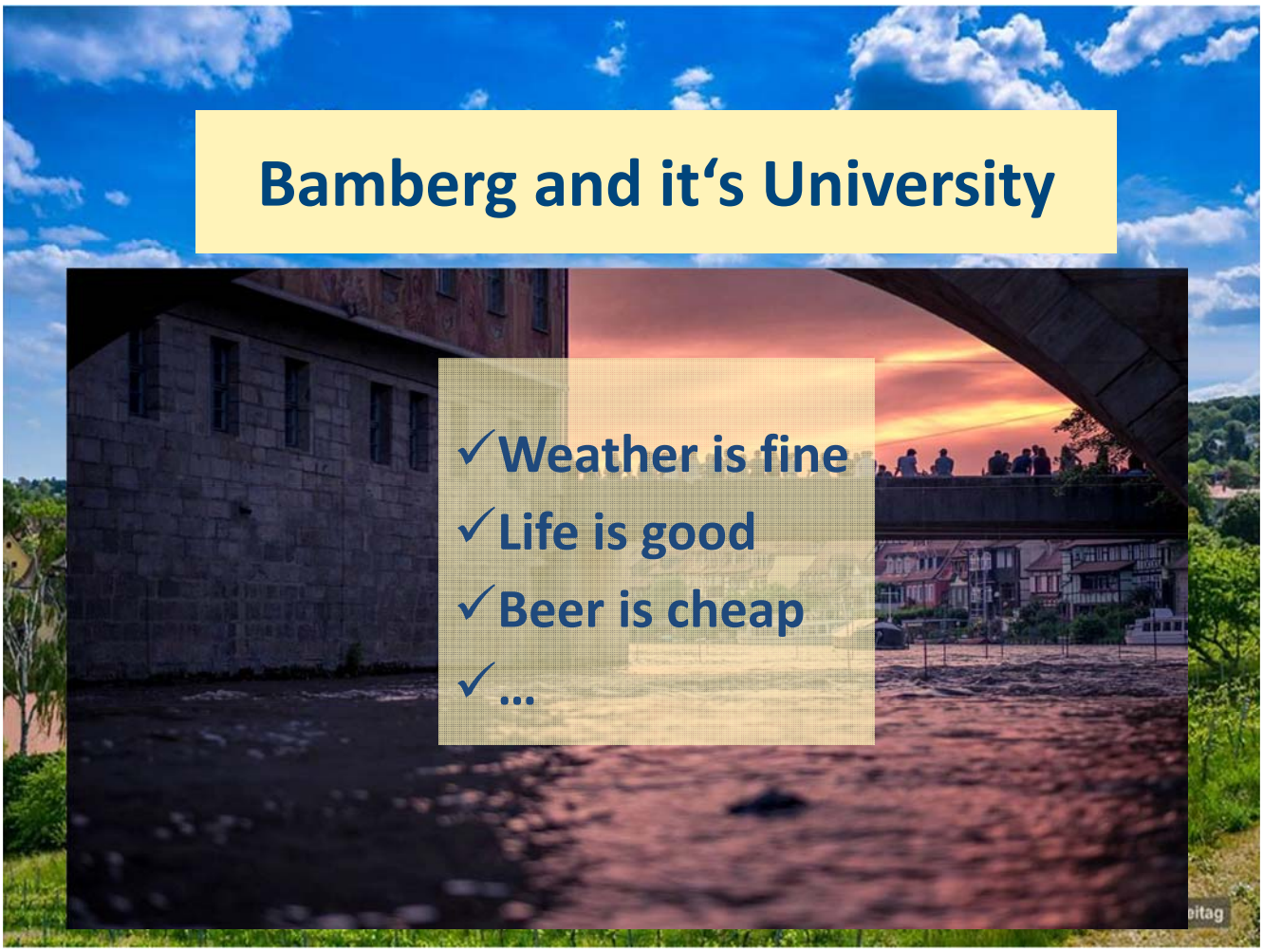
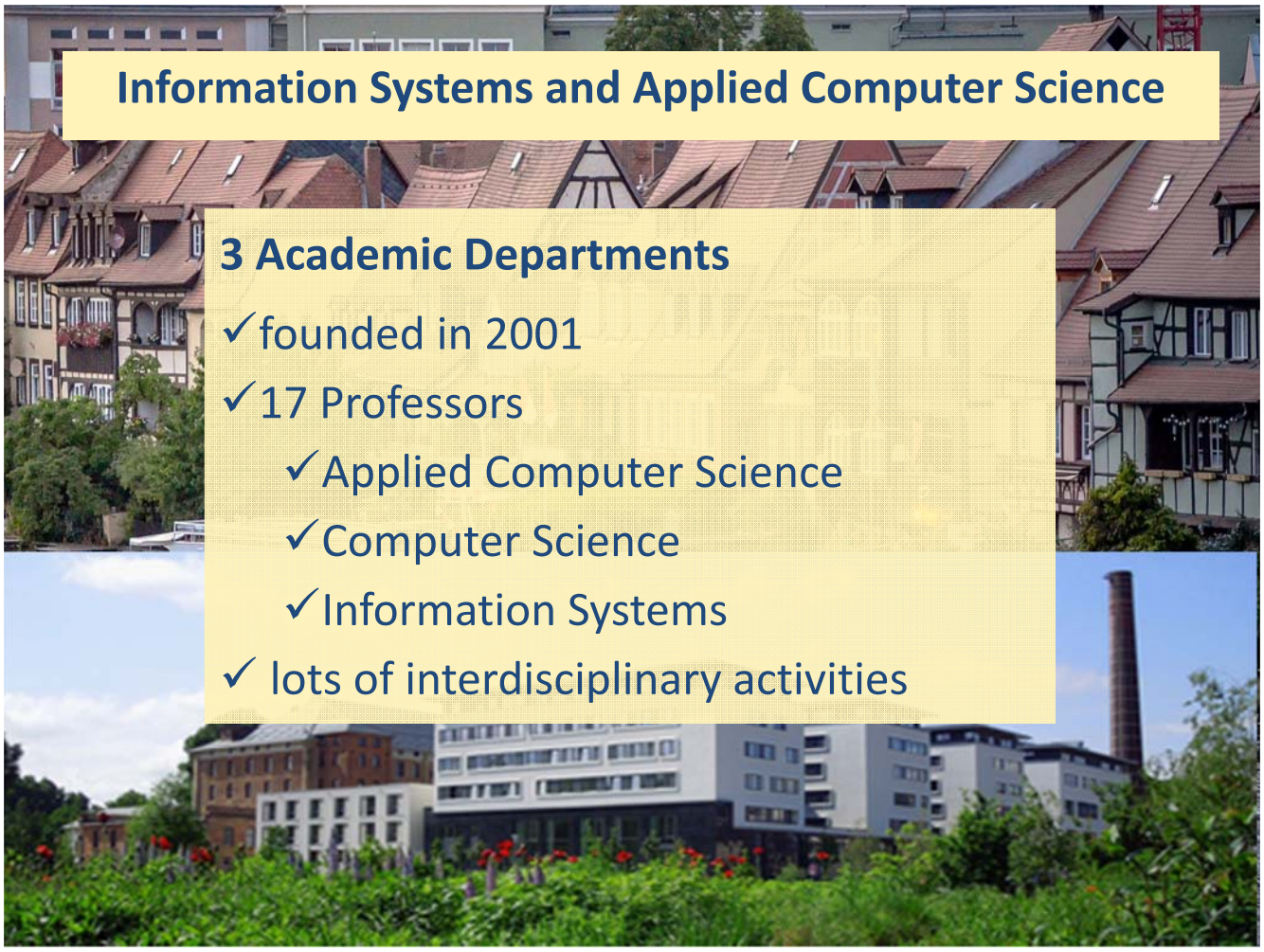
Platform as a Service
– Models and Portability –

11th Summer School
On Service-Oriented Computing
2017

Friday, 2017-06-30, Hersonissos, Crete

Stefan Kolb and Guido Wirtz
Distributed Systems Group
University of Bamberg, Germany





Information Systems and Applied Computer Science

3 Academic Departments

- ✓ founded in 2001
- ✓ 17 Professors
  - ✓ Applied Computer Science
  - ✓ Computer Science
  - ✓ Information Systems
- ✓ lots of interdisciplinary activities

The Distributed Systems Group

We are hiring … PhD students, PostDocs

https://www.uni-bamberg.de/en/pi/

…lopment for complex, esp.

…on all levels. :

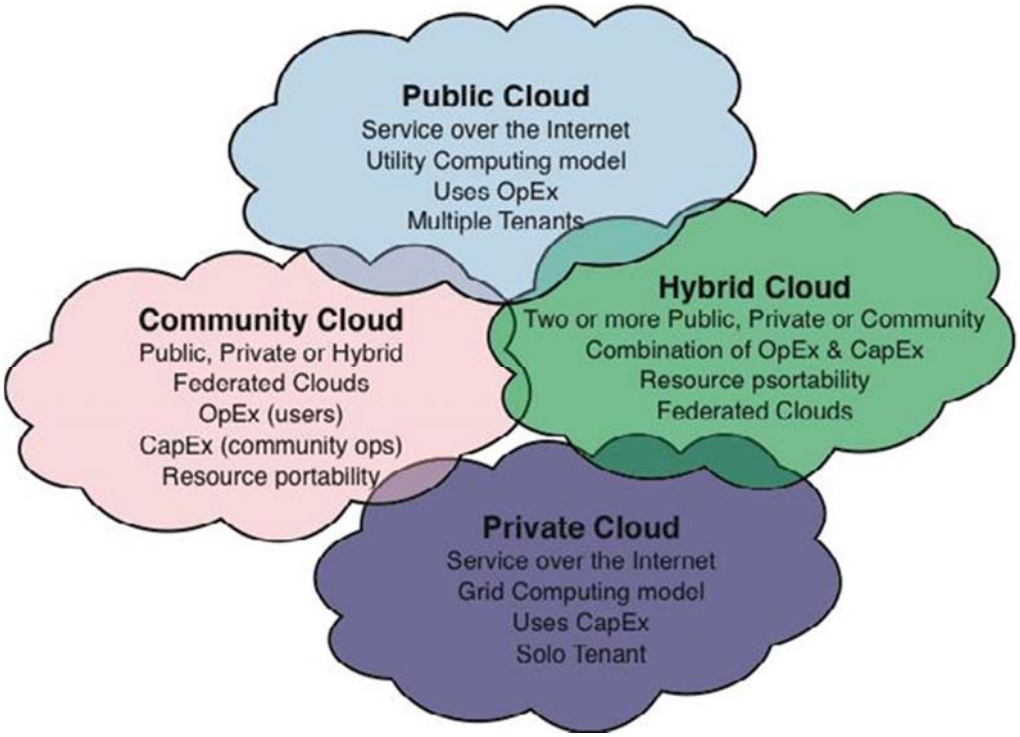
- **…es, service eco systems,**
- **ente… …and B2Bi,**
- **Clouds, esp..**
- **Seamless transitio…**
  their implementation in …
- **Visual design and programming**
  BPMN, BPEL, ….

Talking about Clouds

Def.: "[…] cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction." [NIST 2011]

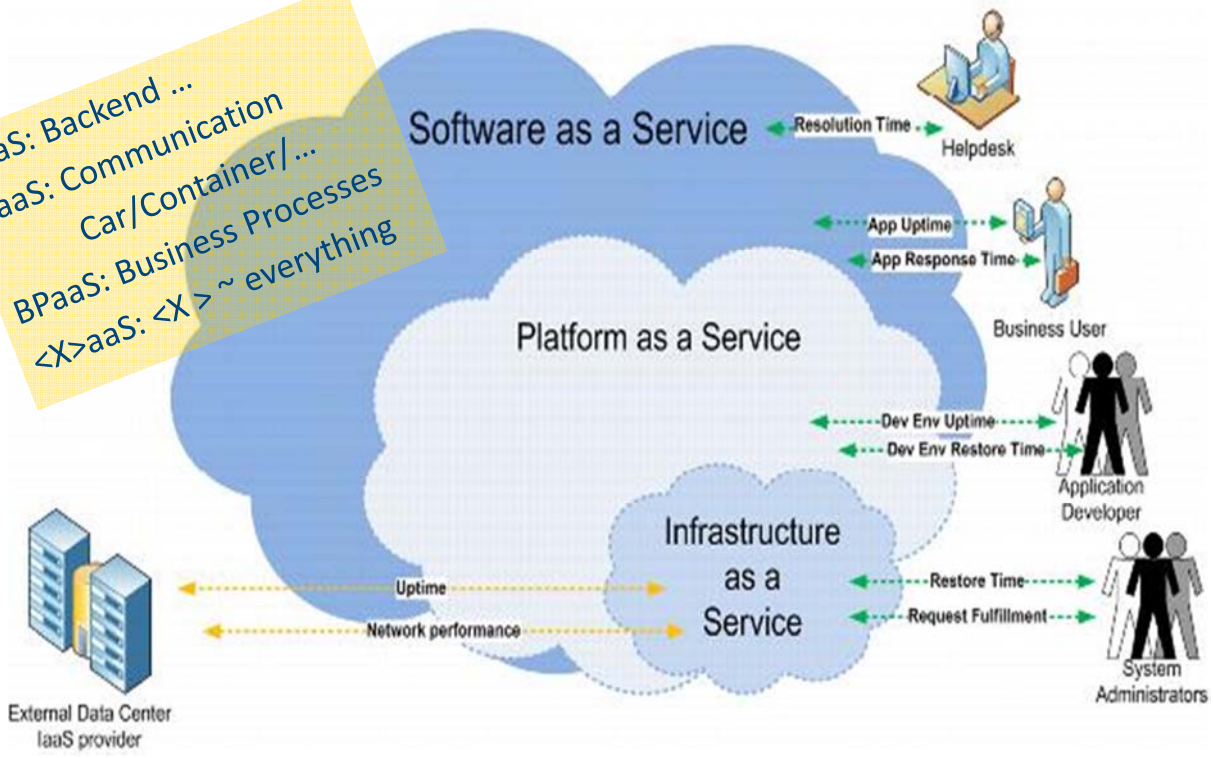
- Five essential Characteristics …
  - on-demand self-service
  - broad network access
  - resource pooling
  - rapid elasticity or expansion
  - measured service

… different deployment models …



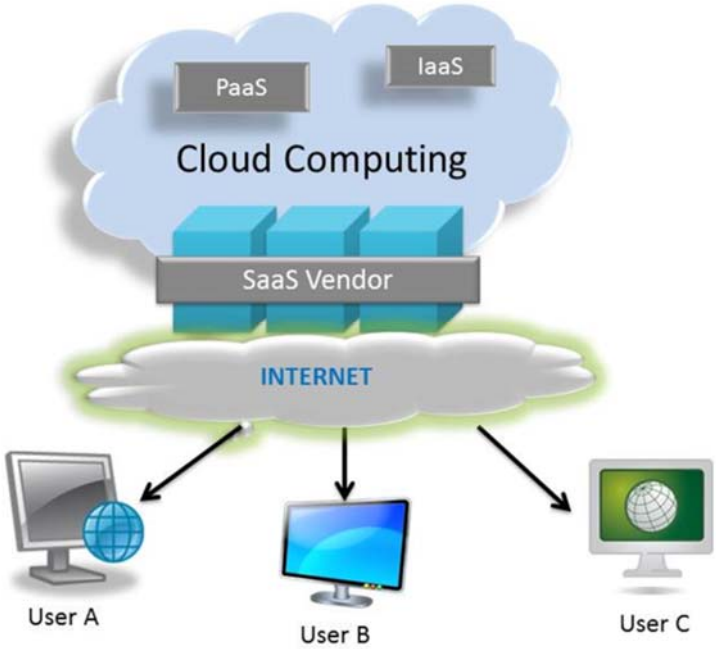
… and a bunch of Service Models …

BaaS: Backend …
CaaS: Communication Car/Container/…
BPaaS: Business Processes
<X>aaS: <X> ~ everything



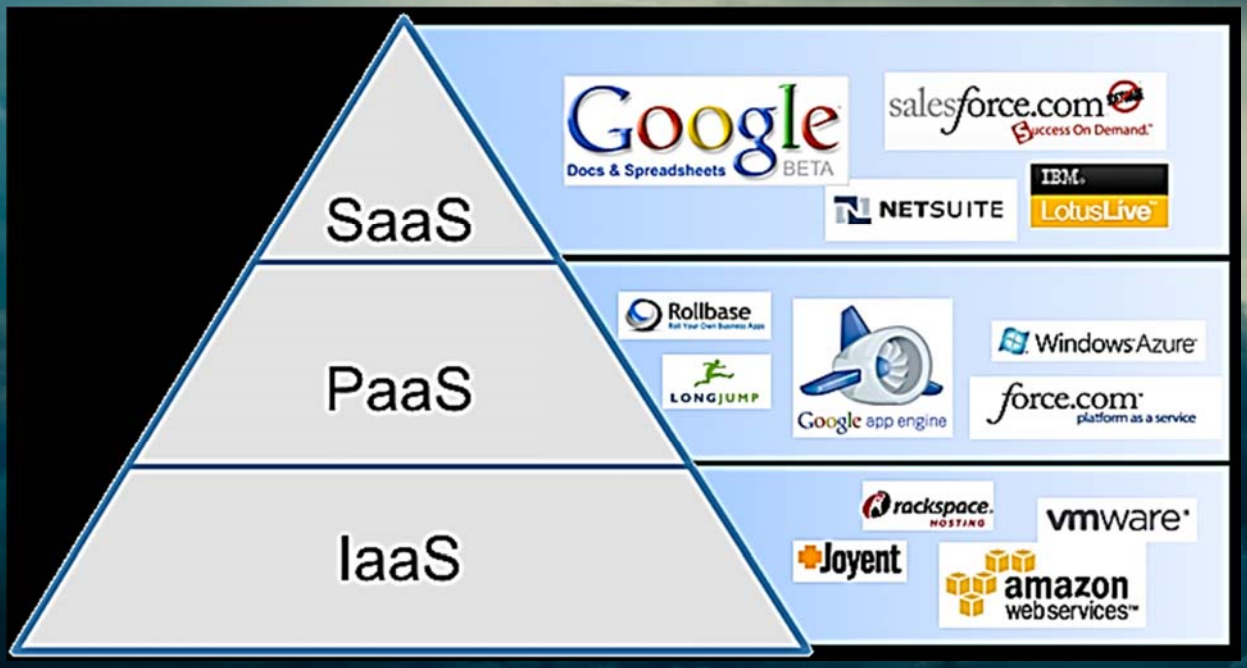
… that are often used in combination

➔‚classical' layered CS model

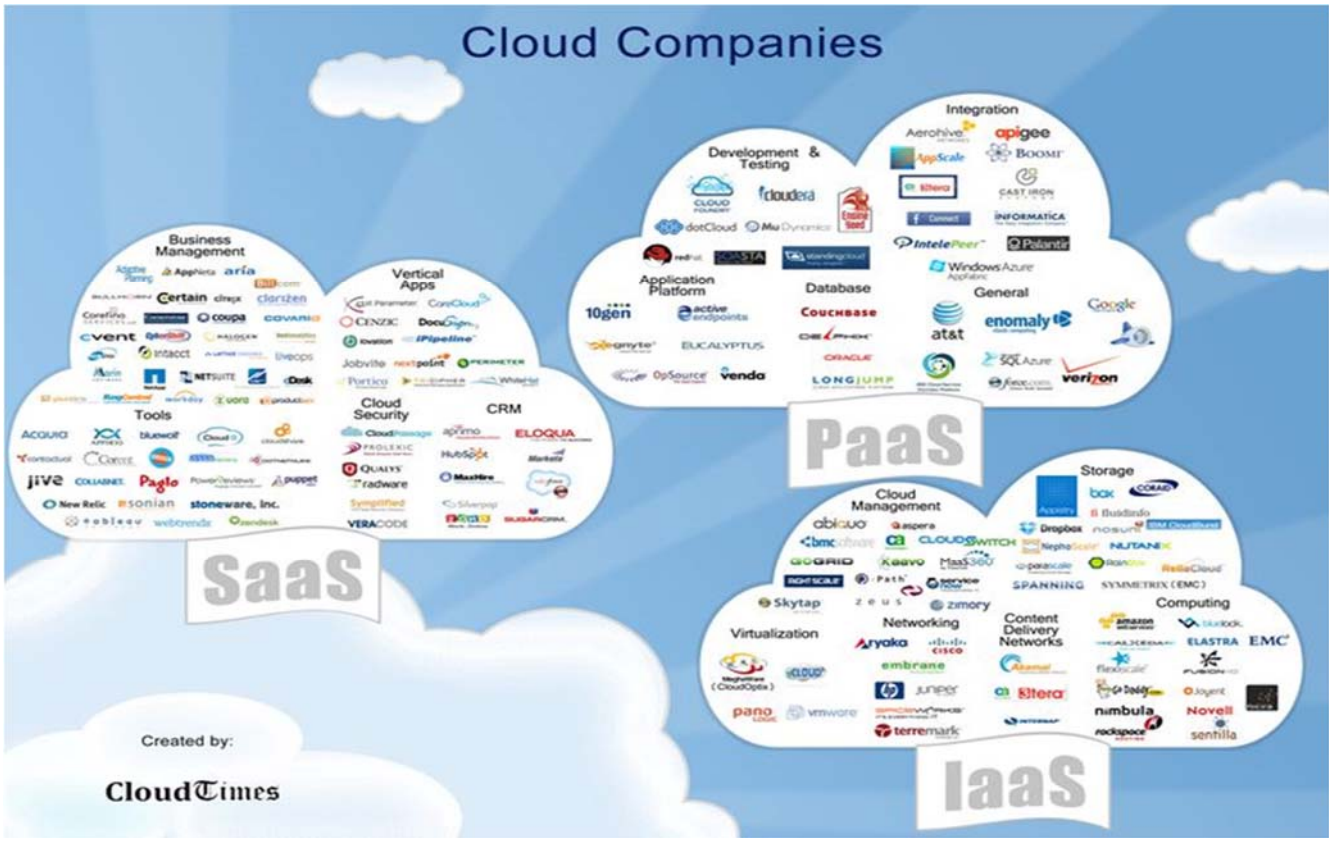


- Abstraction efforts towards the (End-)'User'
- Step-by-step closer to detailed environment when implementing

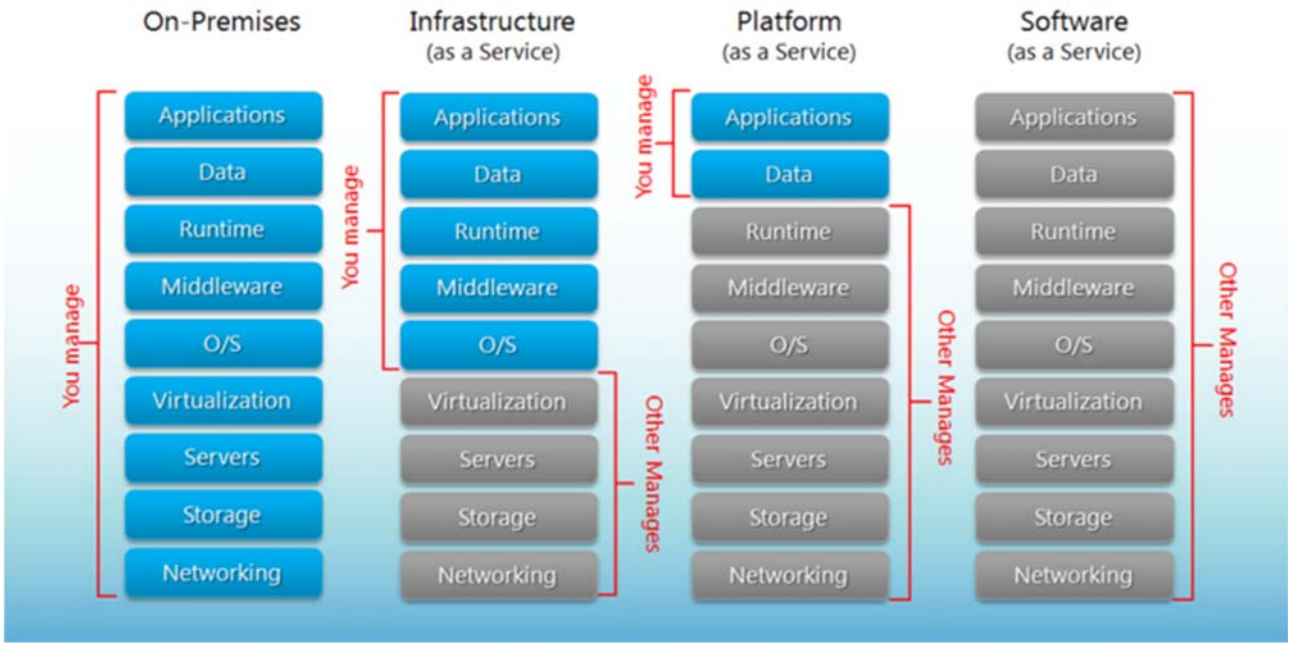
One cloud fits all needs? NO!



Lots of Offers for all Models



A closer Look at Service Models
Separation of Responsibilities



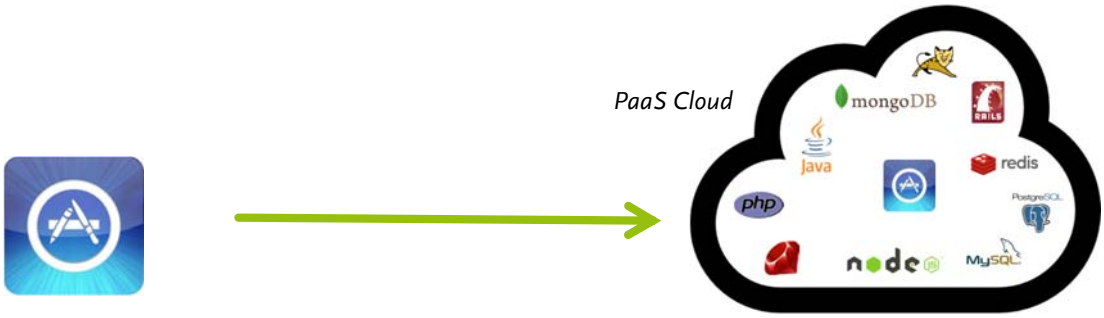
Research and Tutorial Scope

***Platform as a Service (PaaS)*: The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the provider[3]. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly configuration settings for the application-hosting environment. ." [NIST 2011]**

[3]This capability does not necessarily preclude the use of compatible programming languages, libraries, services, and tools from other sources

http://engineyard.co.jp/platform-as-a-service

Why should we focus on PaaS?



- Using an App over the Web ➔ nothing fancy any more
- Providing complex working environments ➔ ???
- Providing ‚Iron' ➔ easy nowadays
  - Compute Centers do that for decades
  - Hardware and basic OS are standard
  - Virtual Machines and Containers

But: Is it important for business, too?

	2016	2017	2018	2019	2020
Cloud Business Process Services (BPaaS)	40,812	43,772	47,556	51,652	56,176
Cloud Application Infrastructure Services (PaaS)	7,169	8,851	10,616	12,580	14,798
Cloud Application Services (SaaS)	38,567	46,331	55,143	64,870	75,734
Cloud Management and Security Services	7,150	8,768	10,427	12,159	14,004
Cloud System Infrastructure Services (IaaS)	25,290	34,603	45,559	57,897	71,552
Cloud Advertising	90,257	104,516	118,520	133,566	151,091
Total Market	**209,244**	**246,841**	**287,820**	**332,723**	**383,355**

Source: Gartner (February 2017)

Remember: SaaS/BPaaS ➔ PaaS ➔ IaaS

Benefits of PaaS Clouds

- Hosted Environment
  - Less configuration than IaaS
  - No hassle with complex distributed systems
  - Instantly become productive for non-standard apps
- Pay for what you need
  - Transparently scale from 1 to 1000s of customers
  - IT grows dynamically with your business

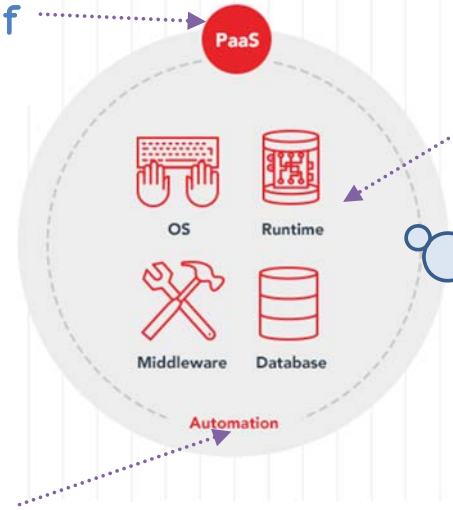
➔ **Focus on Developing & Running Applications**

➔ **Save money**

Issues with the PaaS model

Different notions of PaaS

Heterogeneity among vendors



Lack of standards

What is ‚in the box'?
- Languages, runtimes, versions, …
- Middleware stacks
- Data base models, …

http://engineyard.co.jp/platform-as-a-service

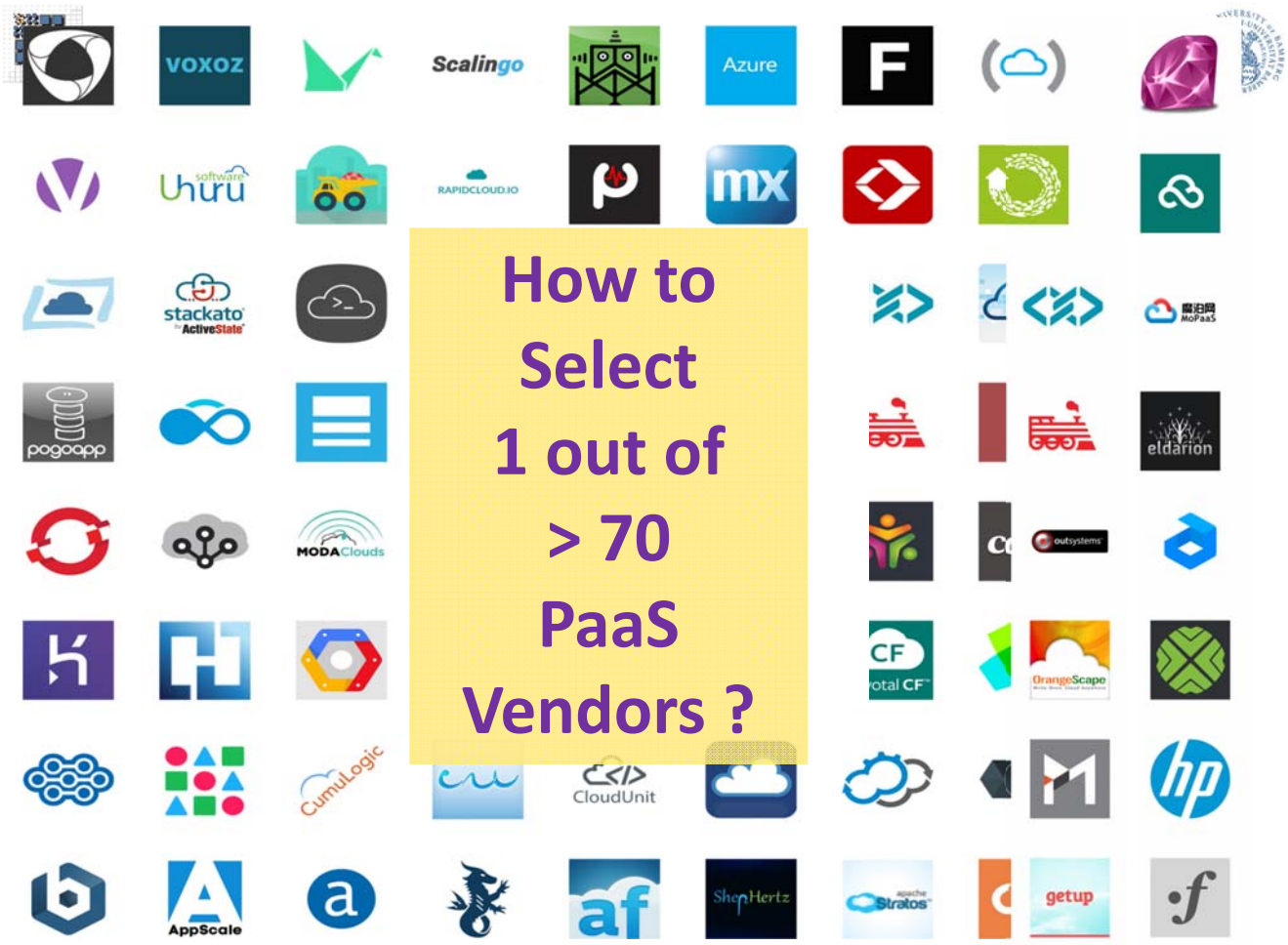
Why is this a problem?

- Developers have different needs
- Software Stacks obtain complex dependencies
- …

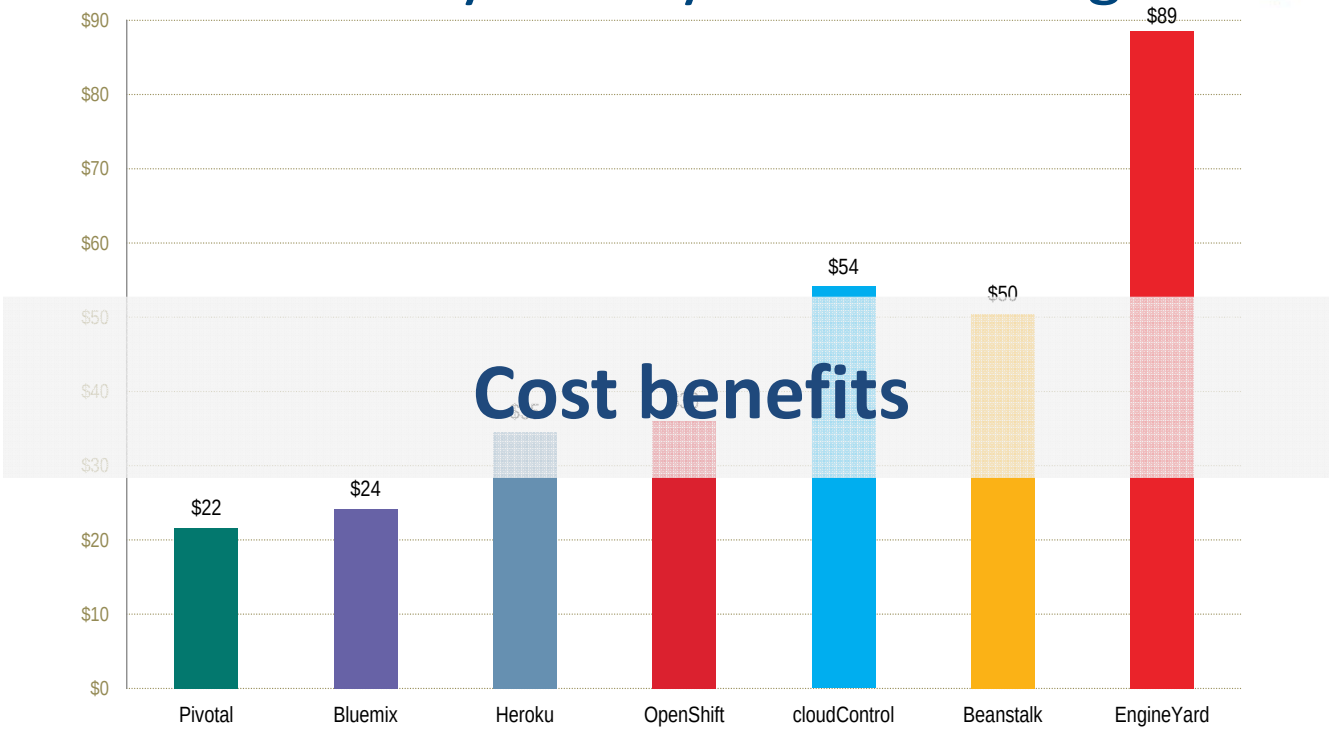
vs.

- Vendors are vague about their offerings
- *„You have to use what you get“*

➔ **How to choose the ‘right’ PaaS platform?**
➔ **What happens if … and you need a new platform?**



Sometimes you may want a change …



Cost benefits

Costs for 2 instances/month OR Costs for 1 VM/month

… and sometimes you have to!

Consolidation & Bankrupcy

Main Issues with current PaaS offers

1. How to choose the ‚right' PaaS Offer?
   - Comprehensible, up-to-date and reliable criteria
   - Offer support for attribute-based selection
2. How to avoid Vendor Lock-In?
   - Try to estimate the prospective effort for porting apps

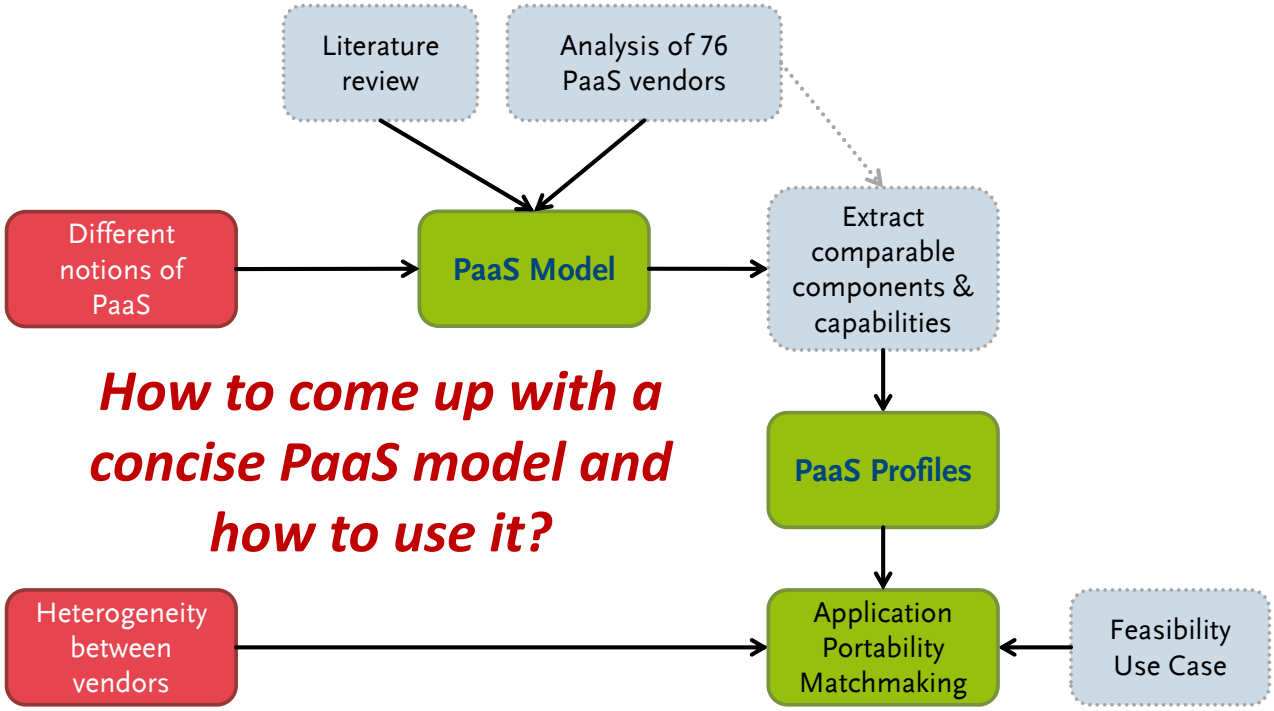
   ➔ **Provide a suitable, operational decision model**
3. How to additionally facilitate Migration ?
   - Concentrate on the most time-consuming issues
   - Abstract from details via a common interface

   ➔ **Provide a common API and it's implementation**

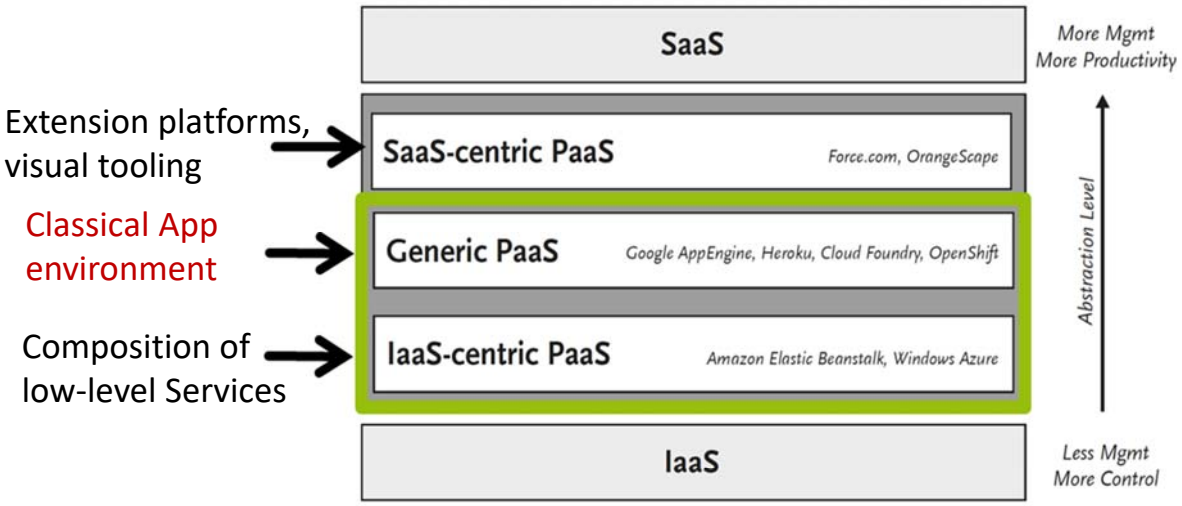
Agenda

I. Introduction and Motivation
II. A Closer Look at PaaS
III. Selecting the ‚right' PaaS
IV. Facilitate Portability Efforts for PaaS
V. Related Work
VI. Conclusion

II. A Closer Look at PaaS



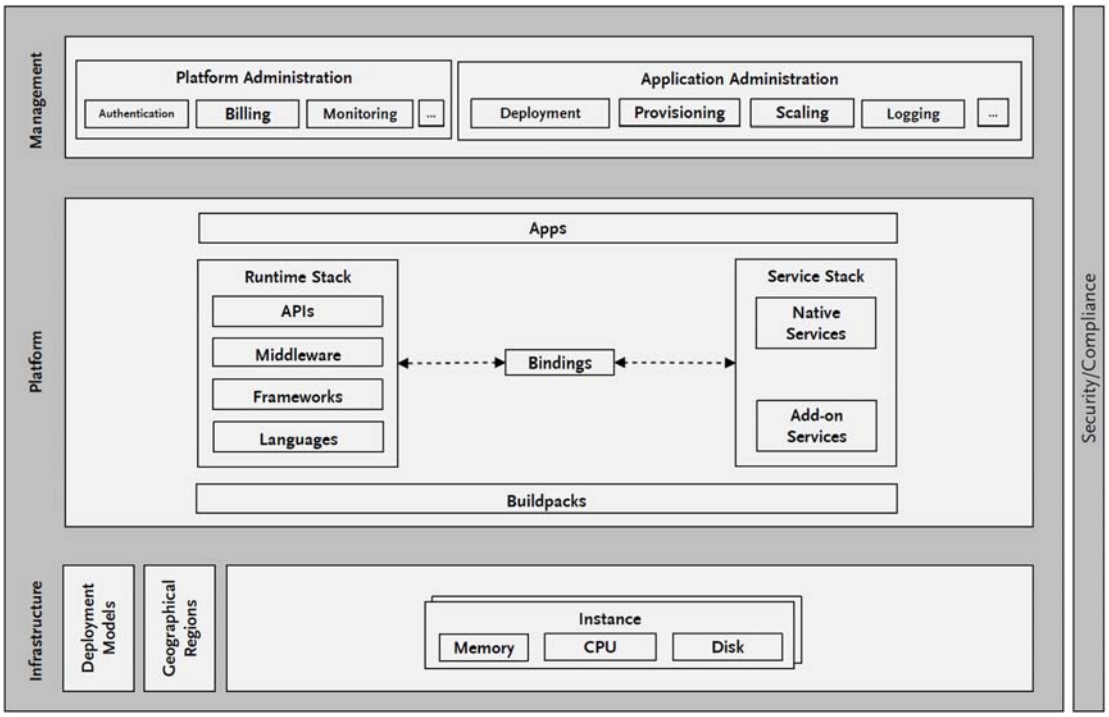
There are three Flavors of PaaS



Comparable application platform & building blocks

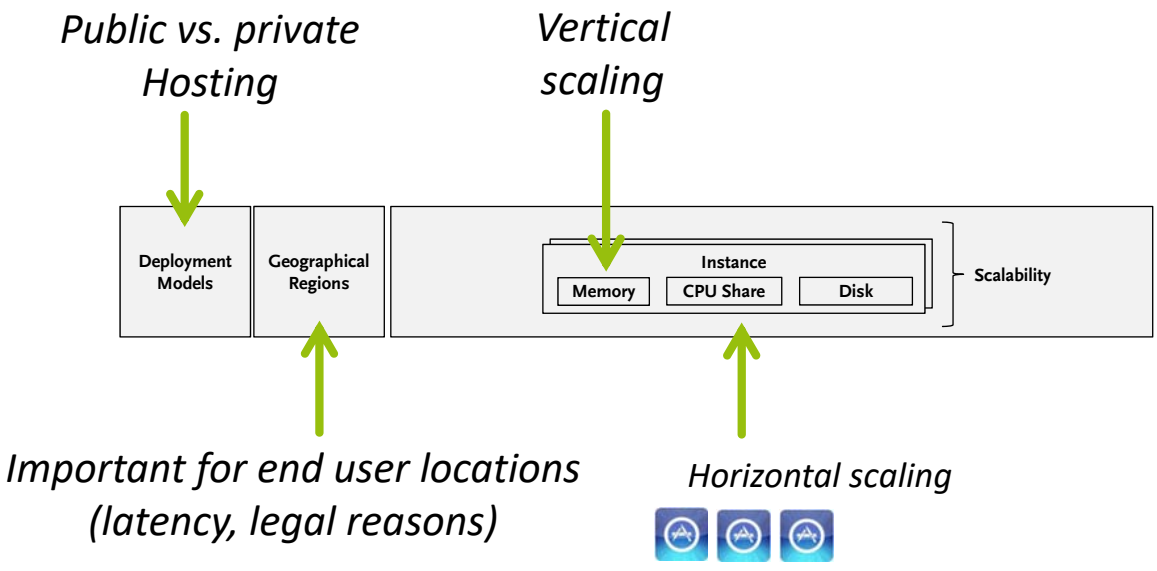
‚Our' PaaS model as a basis for Profiles

Current PaaS ⊆ Model



Infrastructure layer

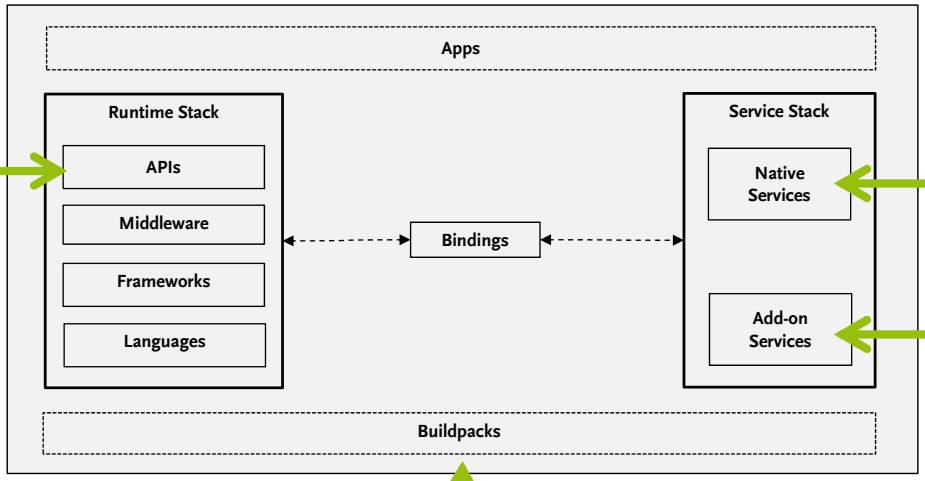
The abstraction of the physical infrastructure or IaaS capabilities.



Platform layer

The service and library interfaces to which the application is written.

⇨ *Application dependency portability*



Including proprietary, platform-specific APIs

Co-located, operated by the PaaS provider

3rd-party providers, SSO, provisioned through PaaS API

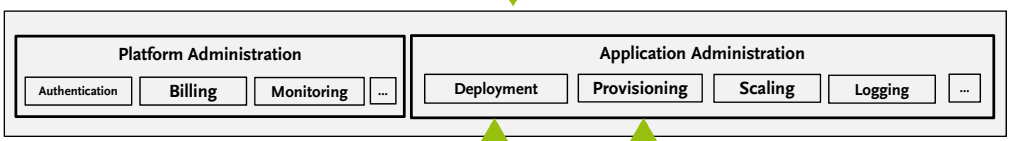
Boundary to IaaS

Extensibility mechanisms for installing missing runtime or service components

Management layer

PaaS users manage their use of the platform.

Accessible via Web-UI, CLI or REST-API



Application life cycle

Services & Add-ons


Agenda

I. Introduction and Motivation
II. A Closer Look at PaaS
III. Selecting the ‚right' PaaS
IV. Facilitate Portability Efforts for PaaS
V. Related Work
VI. Conclusion


Use the Model for PaaS Selection

- PaaS model provides the basis for
  - Extracting attribute sets from vendor data
  - Categorizing PaaS offers from vendors as **PaaS profiles**

  ➔ Searchable Dashboard of > 70 current offers
- ‚Portability Brokering':
  - Analyse Application w.r.t. dependencies and capabilities
  - Describe Needs as a PaaS profile
  - Select suitable PaaS offers through profile matching
  - Decide among resulting set of suitable offers

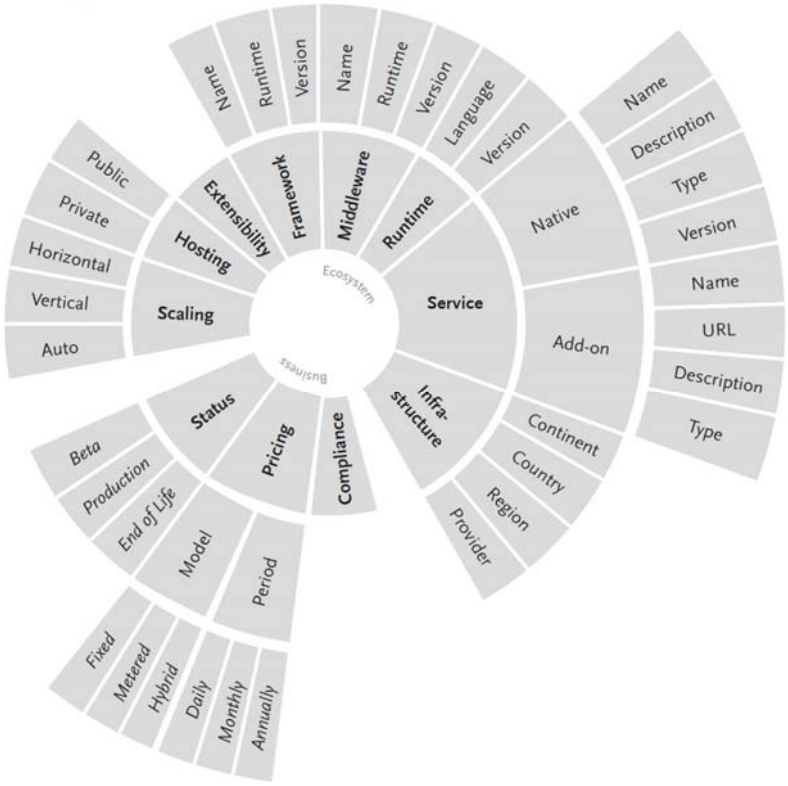
Profile Taxonomy

‚Ecosystem': Components & Capabilities

- Runtimes
- Frameworks
- Middleware
- Services
- Extensibility
- Hosting
- Scaling
- Infrastructures

Business Aspects

- Status
- Pricing
- Compliance



Example: PaaS Profile

Meta-level

Business

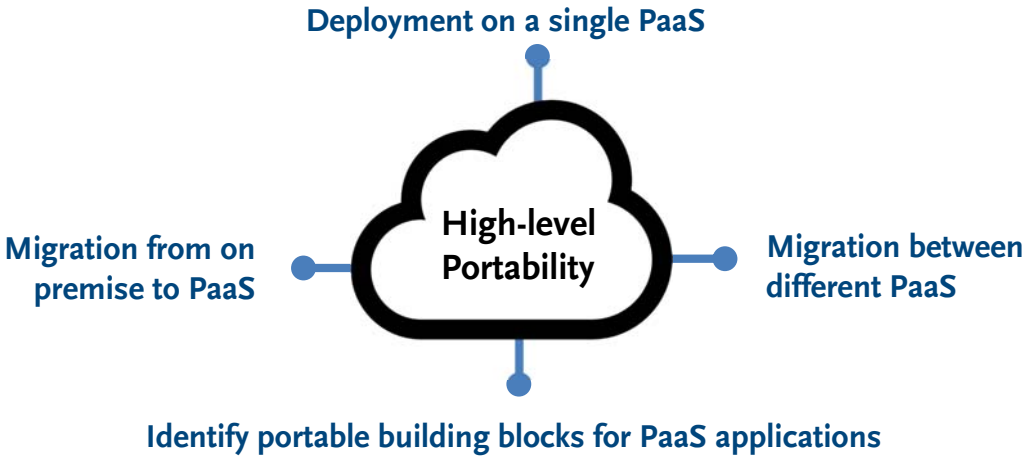
Technical Components & Capabilities

You don't have to read/write this description language

```
{
  "name": "Uniba Paas",
  "revision": "2014-01-26",
  "vendor_verified": "2014-01-26",
  "url": "http://PaaSify.it",
  "status": "production",
  "status_since": "2013-08-01",
  "pricing": [
    {
      "model": "fixed",
      "period": "monthly"
    }
  ],
  "compliance": [
    "SSAE 16 [illegible]",
    "ISAE [illegible]"
  ],
  "scaling": {
    "vertical": true,
    "horizontal": true,
    "auto": false
  },
  "hosting": {
    "public": true,
    "private": false
  },
  "infrastructures": [
    {
      "continent": "NA",
      "country": "US",
      "region": "Virginia",
      "provider": "Amazon Web Services"
    }
  ],
  "runtimes": [
    {
      "language": "java",
      "versions": [
        "1.6", "1.7"
      ]
    }
  ],
  "middleware": [
    {
      "name": "tomcat",
      "runtime": "java",
      "versions": [
        "6.0.35"
      ]
    }
  ],
  "frameworks": [
    {
      "name": "rails",
      "runtime": "ruby",
      "versions": [
        "3.*", "4.*"
      ]
    }
  ],
  "services": {
    "native": [
      {
        "name": "mongodb",
        "description": "Document database",
        "type": "datastore",
        "versions": [
          "2.2"
        ]
      }
    ],
    "addon": [
      {
        "name": "mongohq",
        "url": "https://www.mongohq.com/",
        "description": "MongoDB as a Service",
        "type": "datastore"
      }
    ]
  },
  "extensible": false
}
```

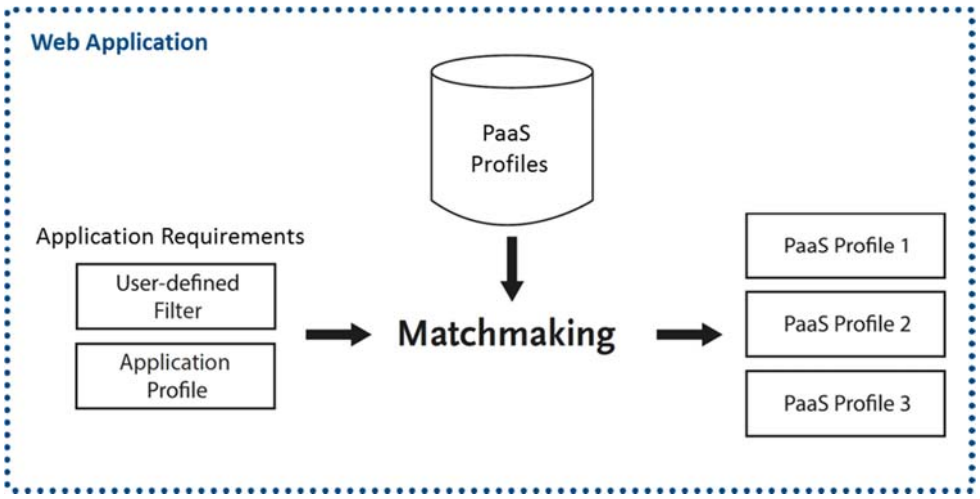
Four Applications for PaaS Profile(s)

- Machine-readable representation of the model
- Profiles serve as description language and catalog



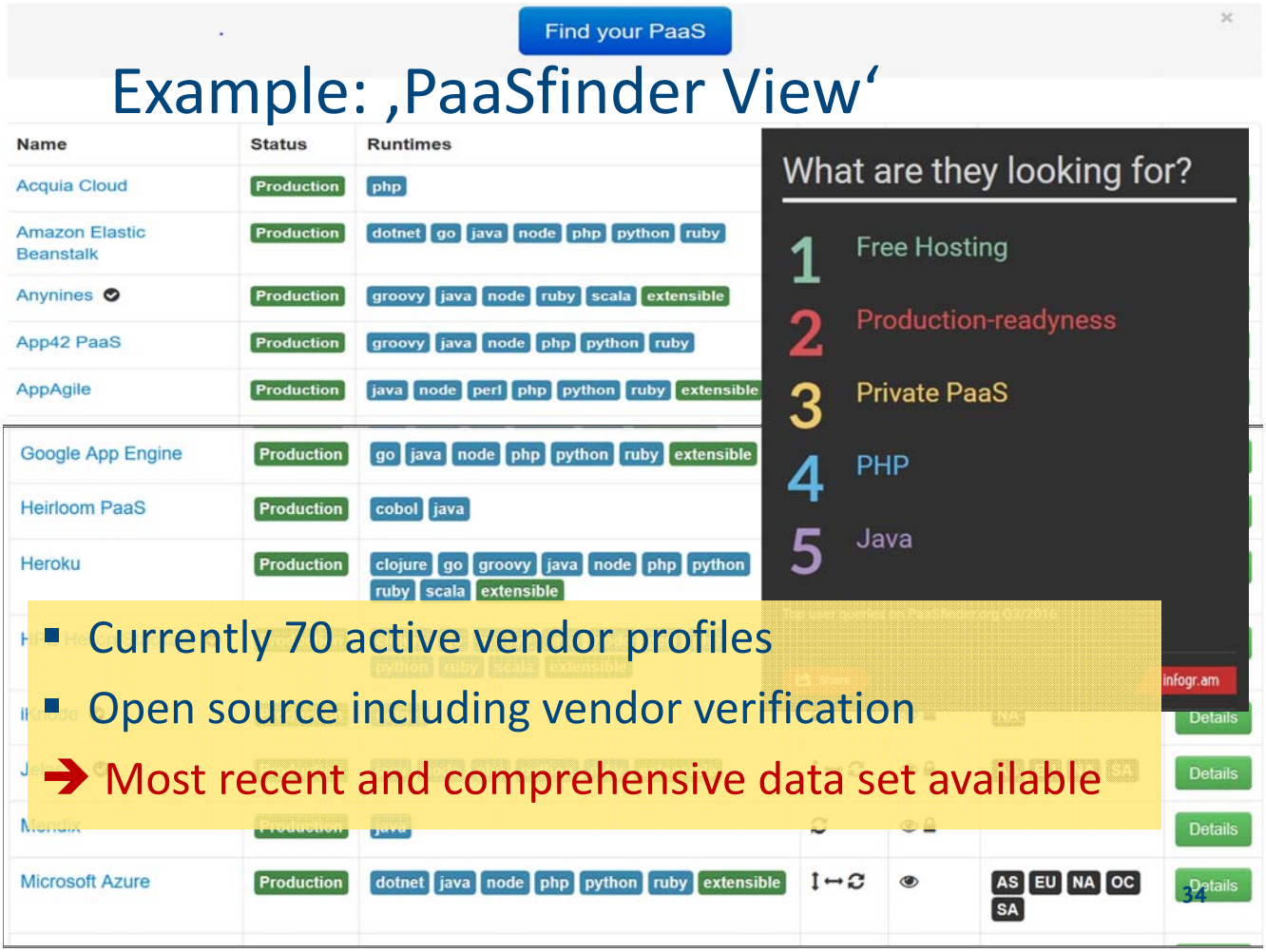
➔ Profile Matching: essential basis for solving PaaS issues

Application Portability Matchmaking



- **User-defined**: Visual selection to satisfy User
- **Application Profile**: Enables automated discovery

Example: ‚PaaSfinder View‘



- Currently 70 active vendor profiles
- Open source including vendor verification

➔ Most recent and comprehensive data set available

Comprehensive

More than 70 vendors

...and counting.

Comparable

Distinctive PaaS features

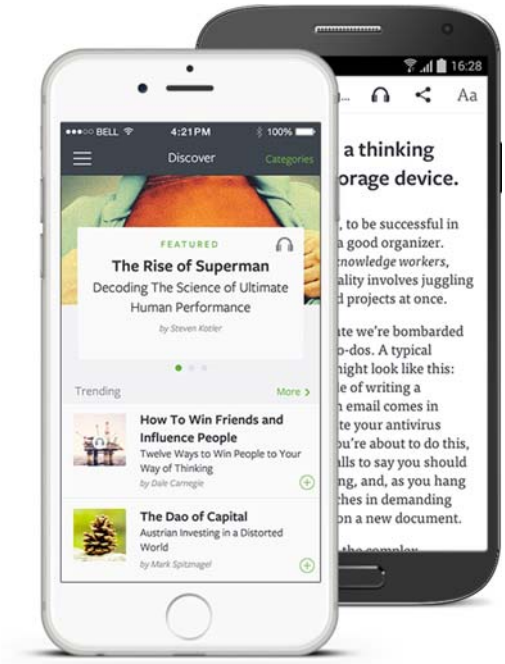
A set of distinctive and intersecting properties to enable comparison and matching of different PaaS offerings.

Current

Continuously updated

Data structures are publicly available and editable by the community. We also aim at vendors to verify their profiles.

Feasibility – Case Study



"Key insights from nonfiction books into fifteen minute reads and audio casts."

Blinkist

1000 books

350 000 registered users

Available for

Android, iPhone, iPad, and **Web**

www.blinkist.com

Download on the App Store | GET IT ON Google play

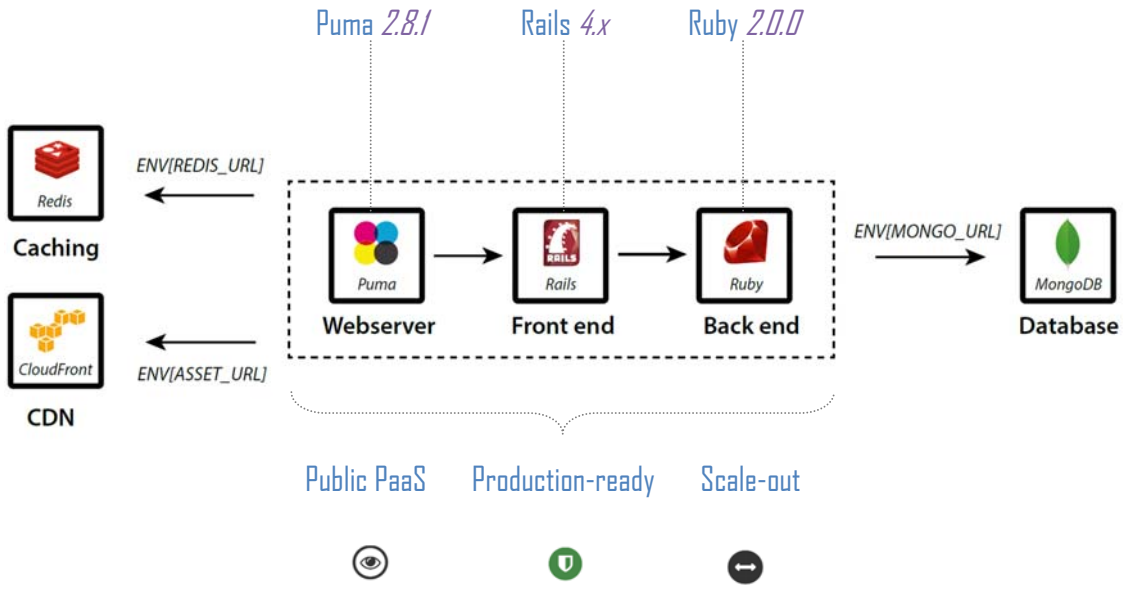
70

vendors

Most recent and comprehensive data set available

http://PaaSfinder.org

Components & Capabilities



Matching components and capabilities

22

vendors

~70 %

ecosystem mismatch

Current Drawbacks & Limitations

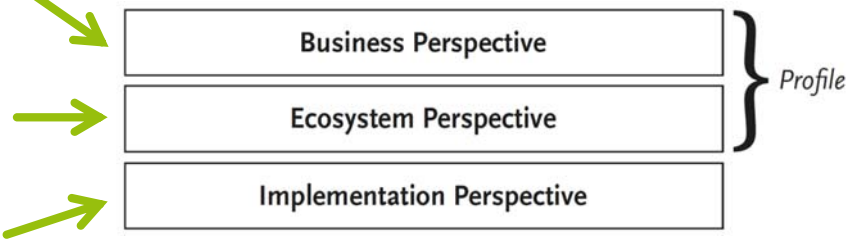
- The model maps well-defined capabilities of the upper two tiers to taxonomies and match them
- Fine-grained implementation details need other checks

Kolb, S., Wirtz, G.: Data Governance and Semantic Recommendation Algorithms for Cloud Platform Selection. in: Proceedings of the 10th IEEE Conference on Cloud Computing, Honolulu, HI, USA, June 25 - June 30, 2017.

Business-relevant capabilities

Technical components and capabilities

Low-level implementation details



- Matching is to be relaxed to 'partial' matching, e.g., in order to take easy to install add-ons or buildpacks into account

➔ **OPEN to suggestions for enhancing the model!**

Agenda


IV. Facilitate Portability Efforts

- Support is
  - Not always possible for all aspects
  - Maybe too costly for a large number of vendors

  ➔ Analyse details of migration costs first
- Concentrate on feasible aspects that
  - reduce considerable efforts
  - are applicable to a wide variety of vendor platforms
  - support agile techniques like DevOps automation

  ➔ Provide API for typical development life-cycle


No Measurement without Metrics!

Based on ISO/IEC SQuaRE Quality Model for code-based metrics

Avg. Deployment Time

Median of a number of deployments

Deployment Reliability

$$\frac{successful\ deployments}{attempted\ deployments}$$

Deployment Effort

Code Changes

LOC of configuration files
+
LOC changes in application code

Build

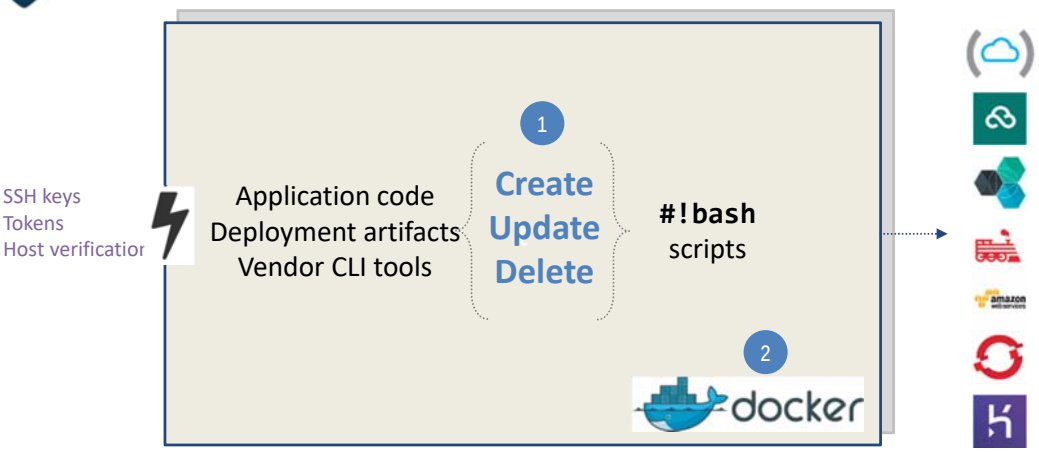
Number of compilation and packaging steps

Steps

Number of deployment steps
+
Number of input parameters

Measurement Automation

1. **Repeatable workflows**
   Prevents measurement errors.
2. **Measurement isolation**
   Local and remote isolation of deployments.



https://github.com/stefan-kolb/paasyard

Measurements

100 runs each

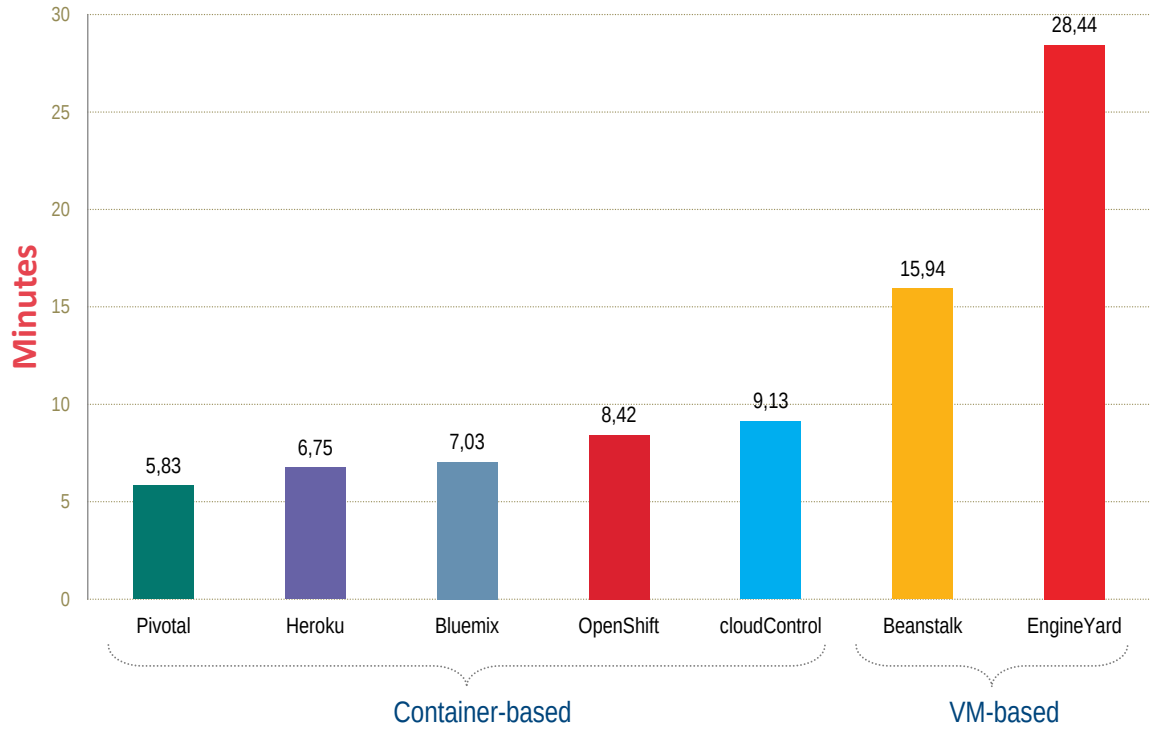
... at varying times during workdays

... in parallel to minimize load-induced influences

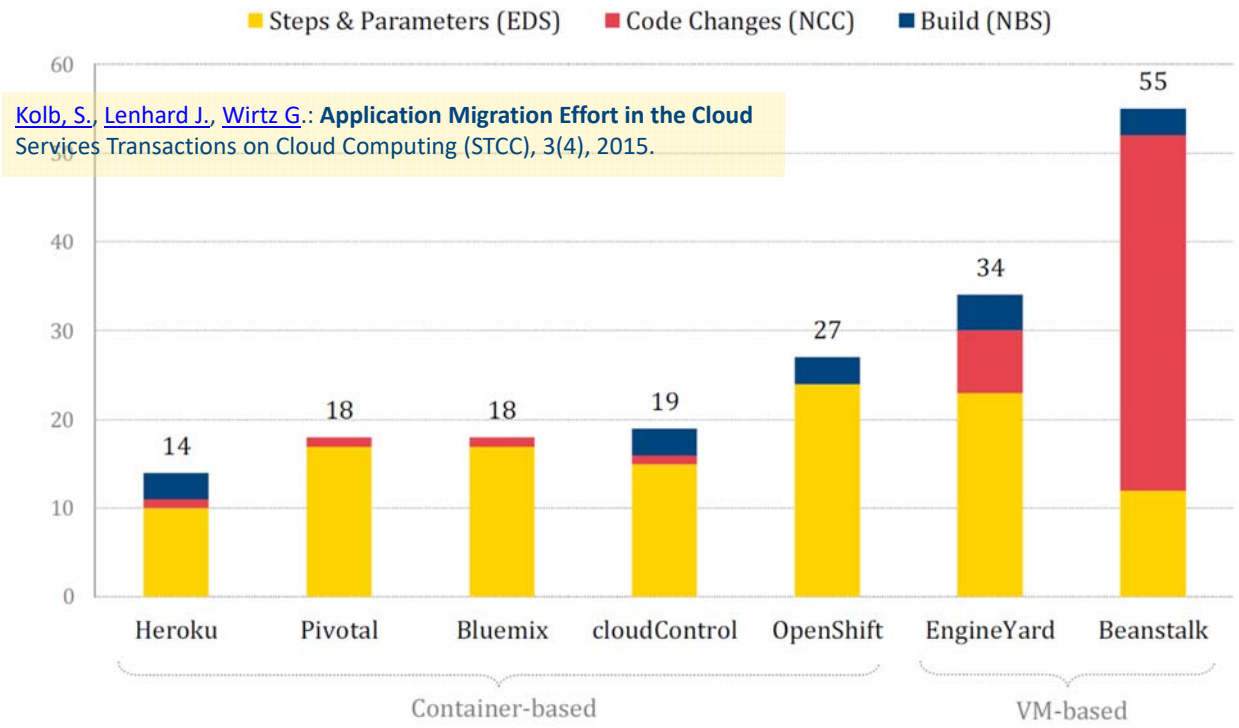
... manually analyze deployment script efforts

Example: Porting Blinkist

Average 1th Time Deployment Time(s)



Deployment Effort is real, varies …



Kolb, S., Lenhard J., Wirtz G.: **Application Migration Effort in the Cloud** Services Transactions on Cloud Computing (STCC), 3(4), 2015.

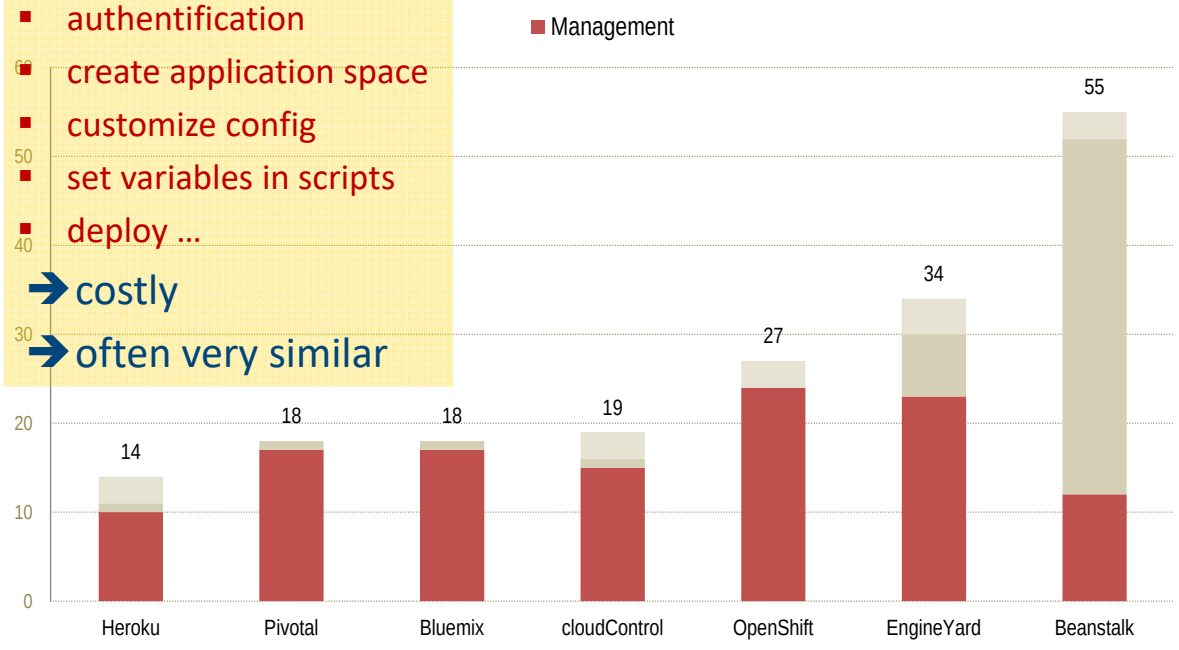
… and consists of lots of Management

Management efforts

- authentification
- create application space
- customize config
- set variables in scripts
- deploy …

➔ costly

➔ often very similar



… which is often the same ‚*in other words*‘

```
heroku auth:login

eb init

heroku create $APPNAME

eb create $APPNAME-env

heroku config:set
${evar[@]}

cf set-env $APPNAME

git push heroku master

cf push $APPNAME
```

```
rhc setup <<EOF
$OS_USERNAME
$OS_PASSWORD
dploy
EOF

rhc create-app
$env(APPNAME) ruby-
2.0

rhc set-env
${evar[@]}

eb setenv ${evar[@]}

git push openshift
master
```

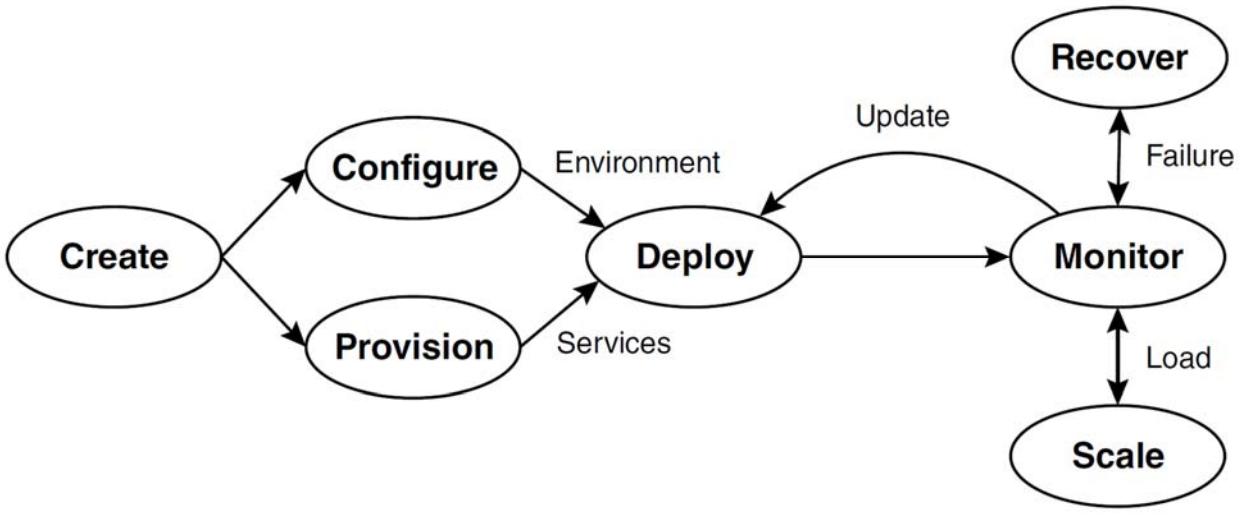
```
cf login -u
"$BM_USERNAME" -p
"$BM_PASSWORD"

cctrlapp
$env(APPNAME)
create ruby

cctrlapp
$APPNAME/default
config.add
${evar[@]}

cctrlapp
$APPNAME/default
deploy
```

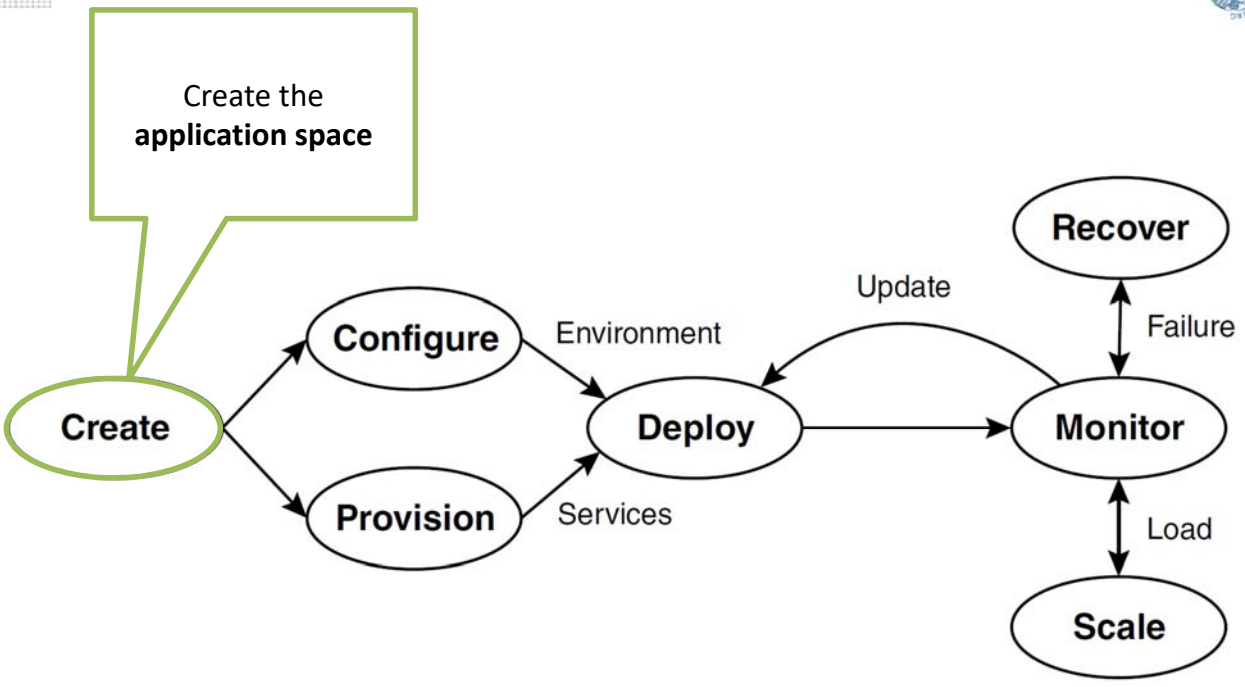

S. Kolb and C. Röck, "Unified Cloud Application Management," in: Proc. World Congress on Services, 2016.

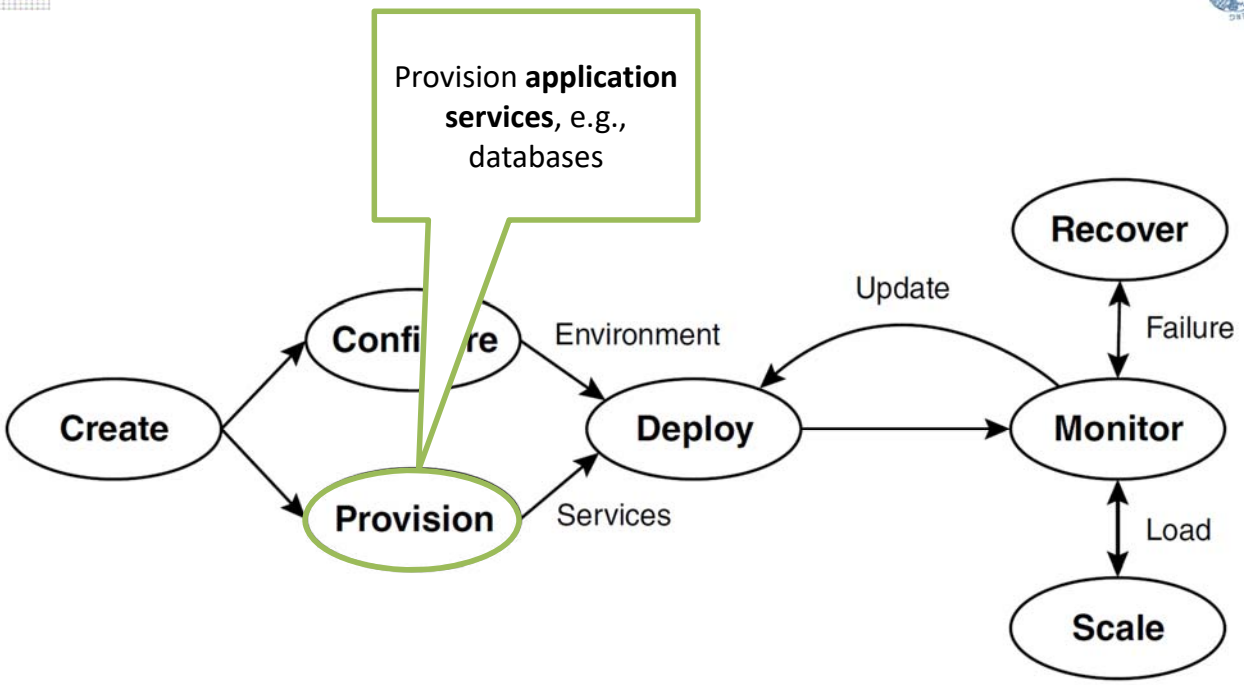
TABLE I: Unified interface operations

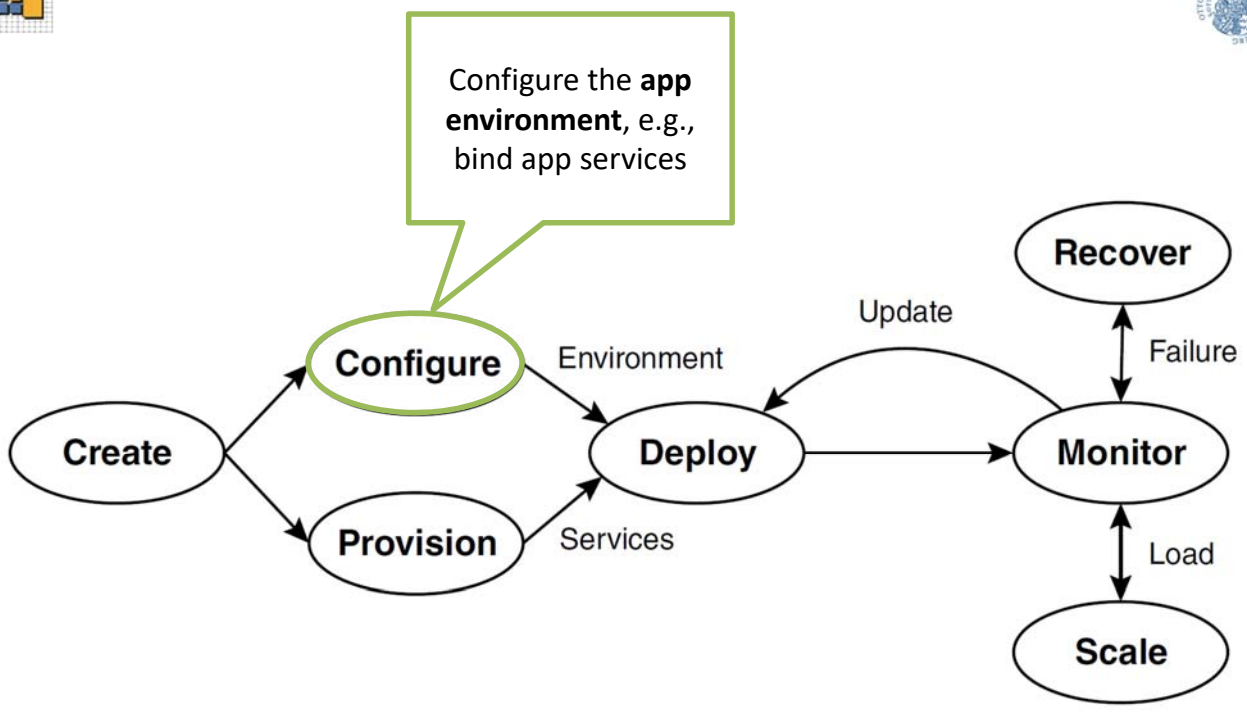
		Functionality	Description	Cloud Foundry v2	Heroku	cloudControl	OpenShift v2	Literature
Application operations	App	GET	Get an app entity	✓	✓	✓	✓	[9], [10], [12]
		DELETE	Delete the app	✓	✓	✓	✓	[9], [10], [12]
		UPDATE	Update the app	✓	✓	✗	✗	[9], [10], [12]
	Lifecycle	REBUILD	Rebuild, e.g. to use updated buildpacks	✓	✓	✓	✓	
		DEPLOY	Upload the actual app data	✓	✓	✓	✓	[9], [10], [12]
		DOWNLOAD	Download the current app data	✓	✓	✓	✓	
		START	Start the app	✓	✓	✗	✓	[9], [10], [12]
		STOP	Stop the app	✓	✓	✗	✓	[9], [10], [12]
		RESTART	Restart the app	✓	✓	✗	✓	[10], [12]
	Scaling	ADD INSTANCE	Add new instance, scale horizontally	✓	✓	✓	✓	[10], [12]
		REMOVE INSTANCE	Remove instance, scale horizontally	✓	✓	✓	✓	[10], [12]
		SCALE INSTANCE	Set instance power level	✓	✓	✓	✗	
	Domains	LIST DOMAINS	List all the app's domains	✓	✓	✓	✓	
		GET	Get domain entity	✓	✓	✓	✓	
		ADD DOMAIN	Assign domain to the app	✓	✓	✓	✓	
		DELETE	Delete and remove the domain	✓	✓	✓	✓	
		UPDATE	Update the domain settings	✓	✓	✓	✓	
	Variables	LIST VARS	List all env. variables of the app	✓	✓	✓	✓	
		CREATE VAR	Create a variable with initial value	✓	✓	✓	✓	
		UPDATE VAR	Update an existing variable's value	✓	✓	✓	✓	
		DELETE VAR	Remove a variable	✓	✓	✓	✓	
		GET VAR	Get an environment variable entity	✓	✓	✓	✓	
	Logging	LIST LOGS	Collect the app's logfiles	✓	✓	✓	✓	[10]
		GET SPECIFIC LOG	Get a specific log file	✓	✓	✓	✓	
		DOWNLOAD LOGS	Download all logs as archive	✓	✓	✓	✓	
	Services	ADD SERVICE	Install and bind to the app	✓	✓	✓	✓	[9], [10]
		UPDATE SERVICE	Update bound service settings	✓	✓	✓	✓	
		REMOVE SERVICE	Remove bound service	✓	✓	✓	✓	[9], [10]
		GET	Get bound service entity	✓	✓	✓	✓	
		LIST	List all installed services	✓	✓	✓	✓	
General	App	CREATE	Create the app	✓	✓	✓	✓	[9], [10], [12]
		LIST	List all applications	✓	✓	✓	✓	[9], [10], [12]
	Service	GET	Get available service entity	✓	✓	✓	✓	[10]
		LIST	List all available services	✓	✓	✓	✓	[9], [10]
		GET PLAN	Get service plan entity	✓	✓	✓	✓	
		LIST PLANS	List all available plans for a services	✓	✓	✓	✓	
	Region	GET	Get available region entity	✗	✓	✗	✓	
		LIST	List all available regions	✗	✓	✗	✓	
				95 %	100 %	84 %	95 %	

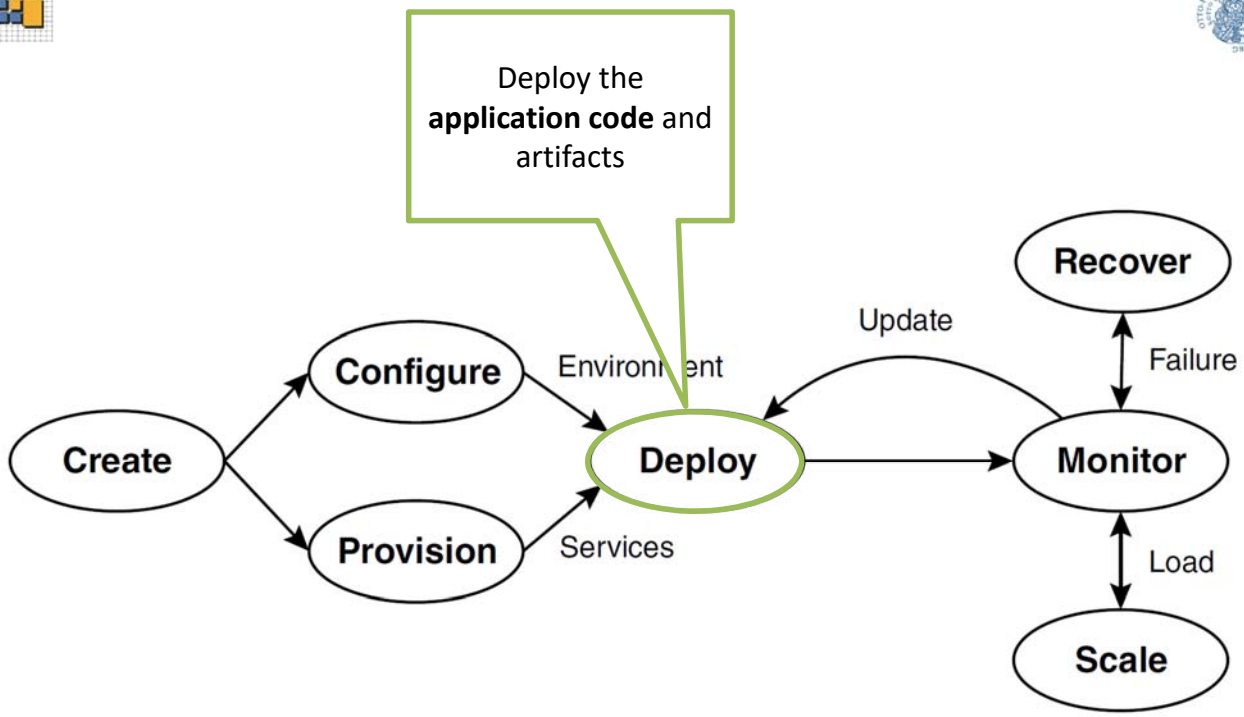
Abstraction: Unified Interface Lifecycle

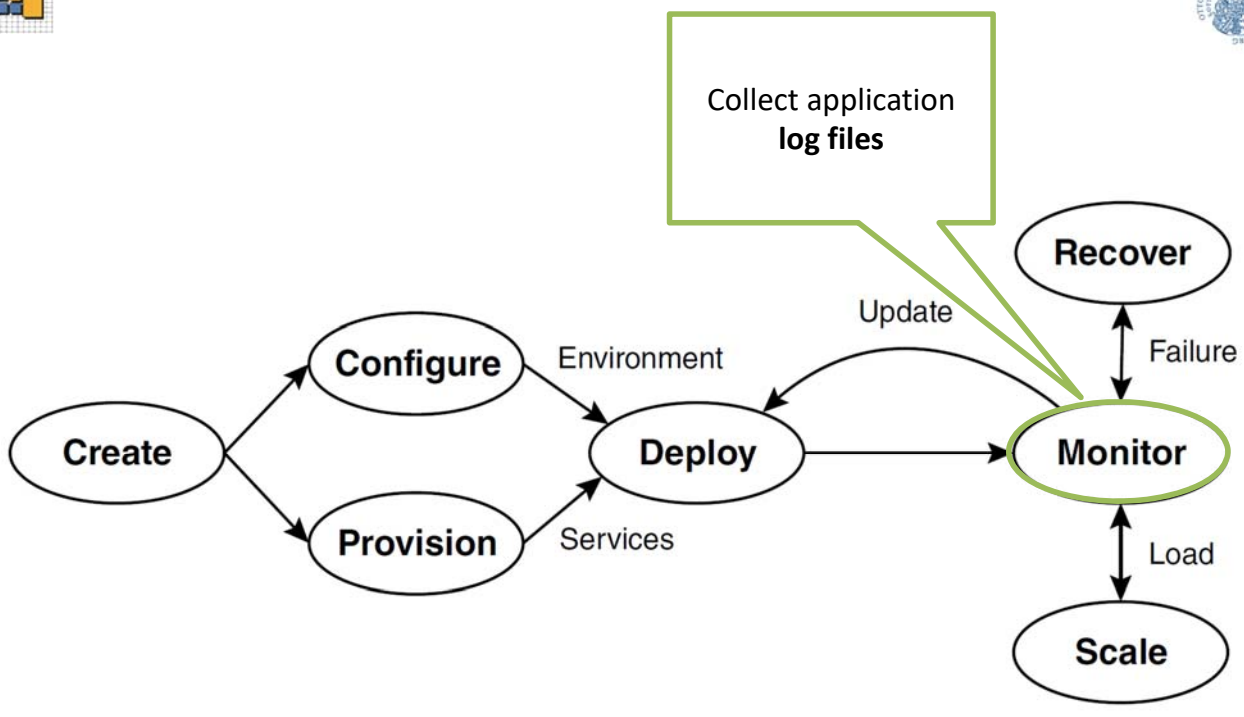


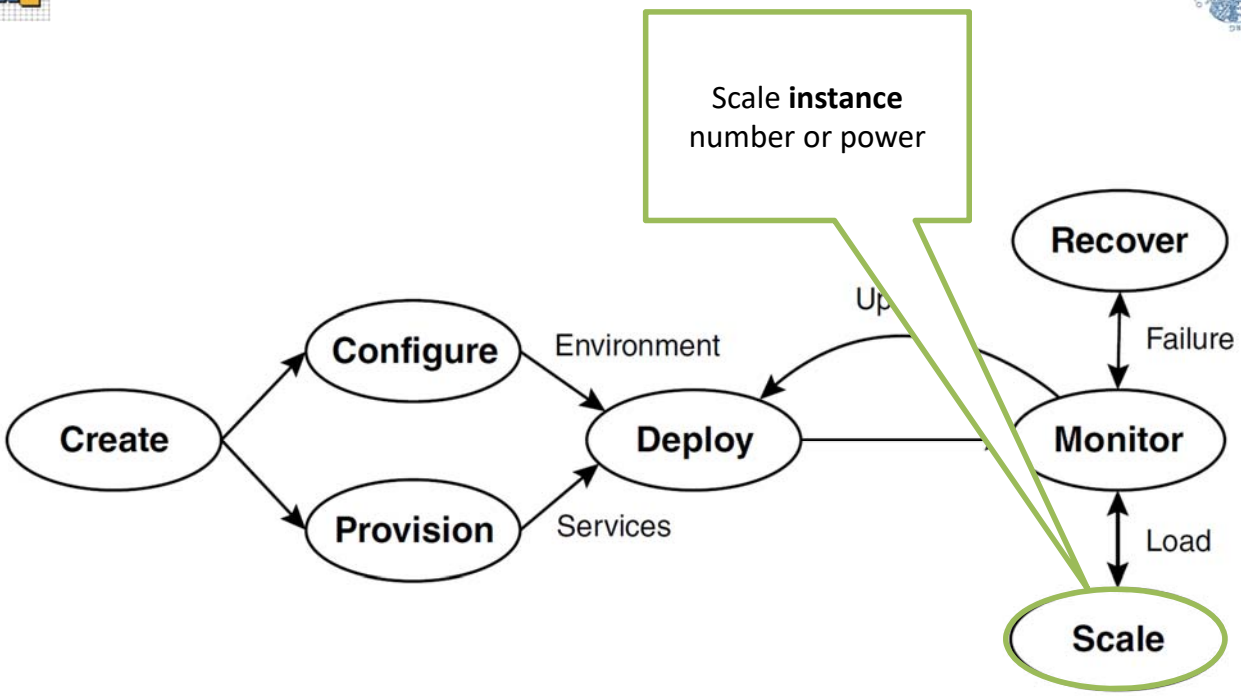


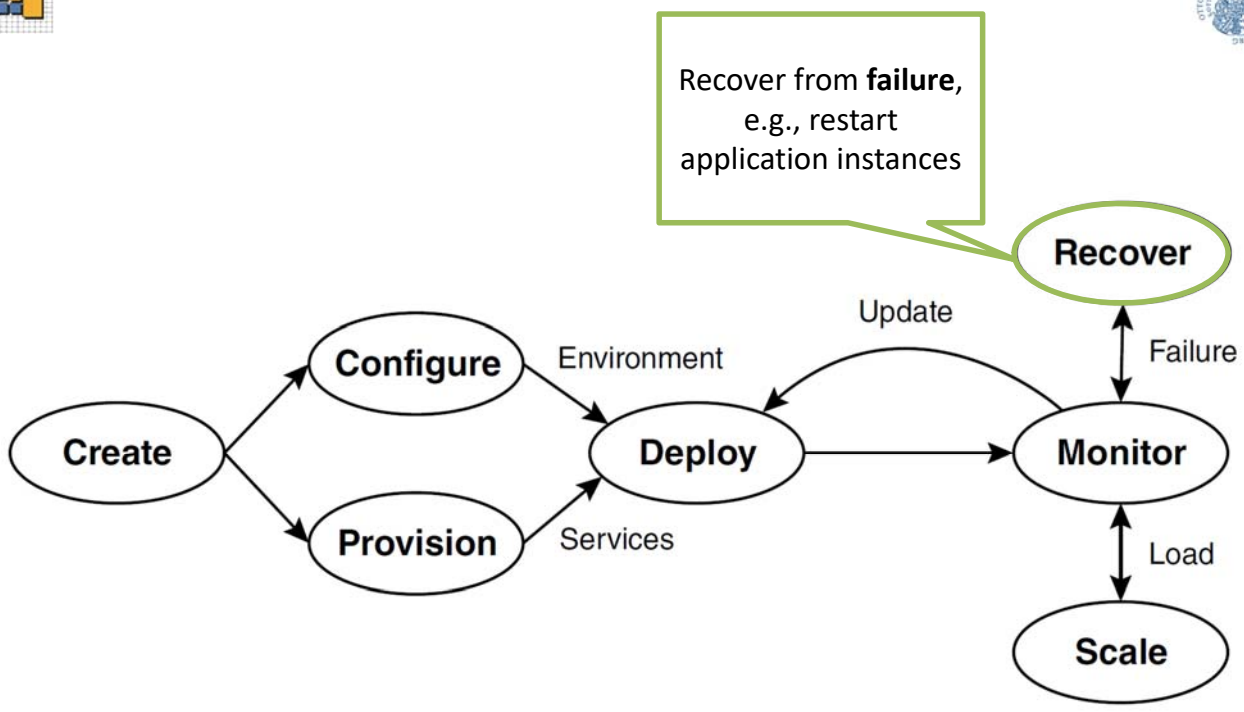

Provision **application services**, e.g., databases
Recover
Update
Failure
Confi re
Environment
Create
Deploy
Monitor
Provision
Services
Load
Scale

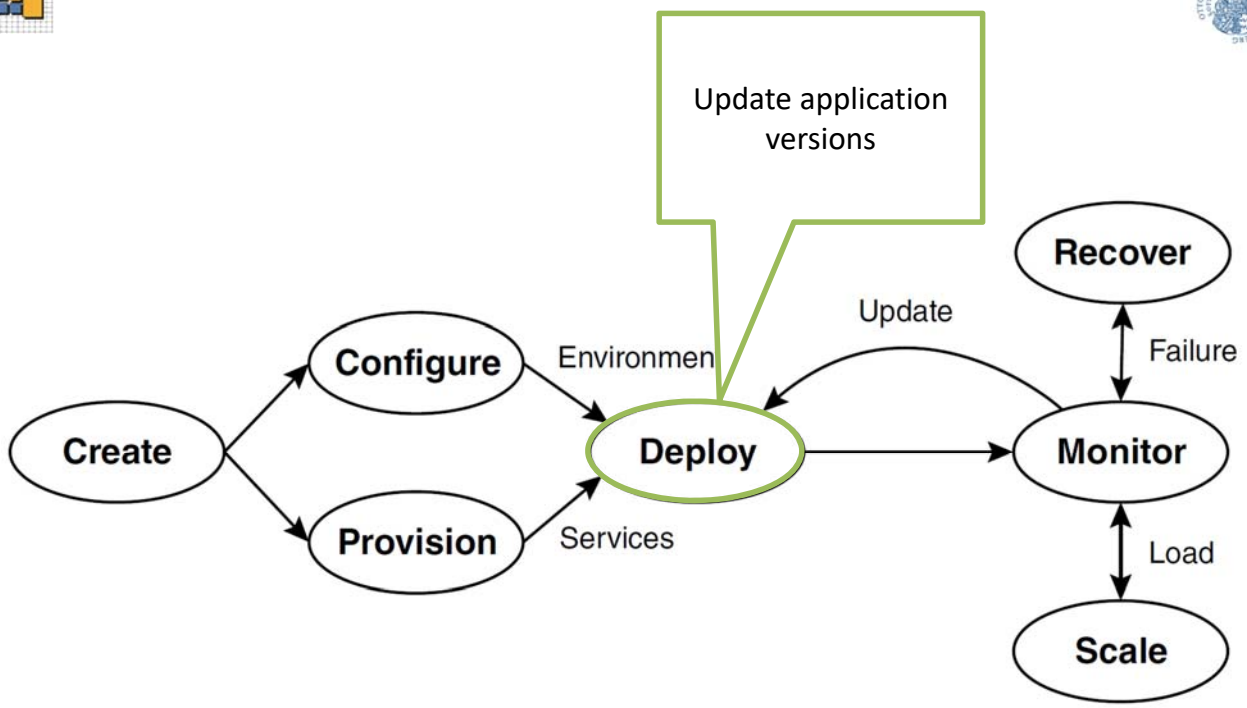

Configure the **app environment**, e.g., bind app services
Recover
Update
Failure
Configure
Environment
Create
Deploy
Monitor
Provision
Services
Load
Scale


Deploy the
application code and
artifacts
Recover
Update
Failure
Configure
Environment
Create
Deploy
Monitor
Provision
Services
Load
Scale


Collect application
log files
Recover
Update
Failure
Configure
Environment
Create
Deploy
Monitor
Provision
Services
Load
Scale


Scale **instance** number or power
Recover
Failure
Monitor
Load
Scale
Configure
Environment
Create
Deploy
Provision
Services


Recover from **failure**, e.g., restart application instances
Recover
Update
Failure
Monitor
Load
Scale
Configure
Environment
Create
Deploy
Provision
Services



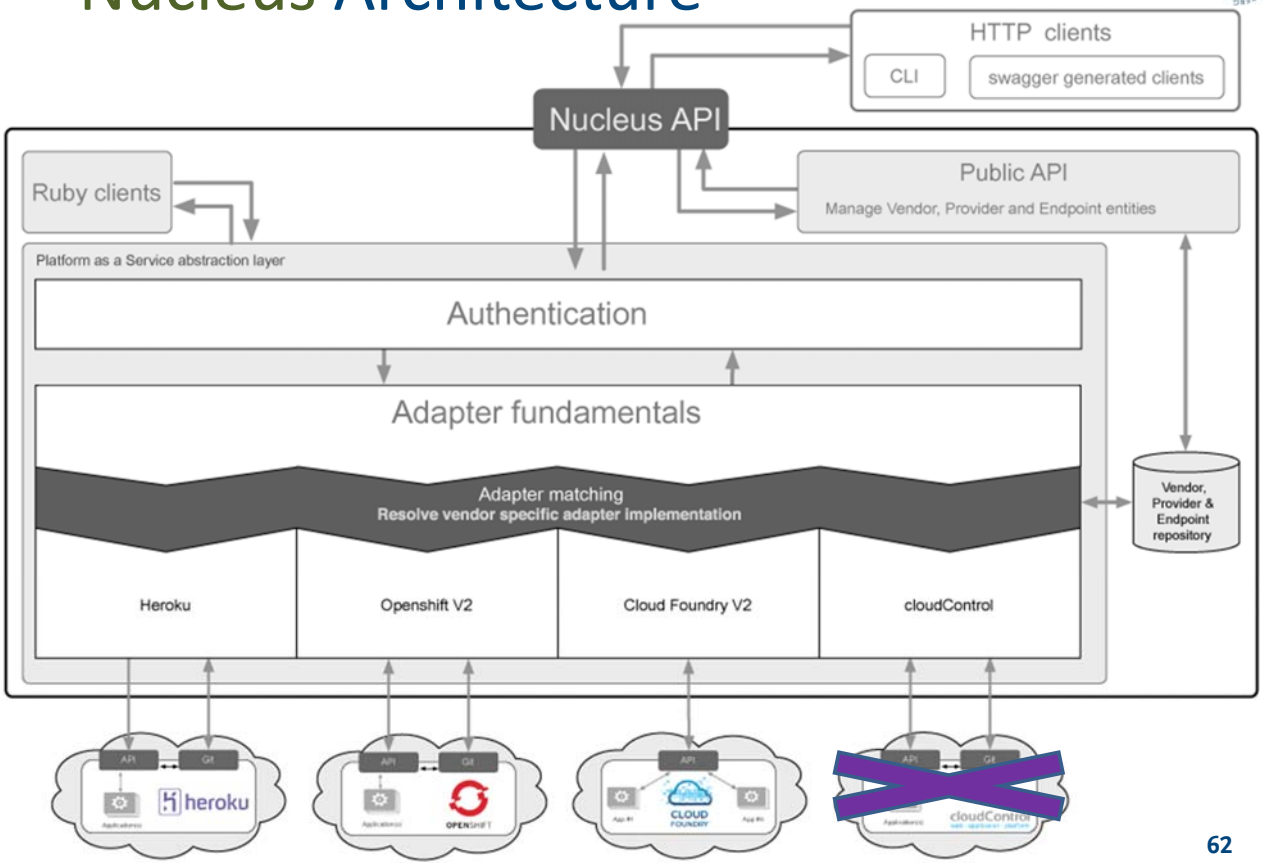
Implementing the IF: Nucleus

- supports currently three important vendors …

Heroku **Cloud Foundry** **OpenShift**

- … plus additional providers from their ecosystem

Nucleus Architecture



Nucleus Features

- Unified management interface
- Language-independent API
- Extensible and versionable
- Fallback to native API

Nucleus

https://github.com/stefan-kolb/nucleus

Unified interface support

Heroku	Cloud Foundry	OpenShift	cloudControl
100 %	100 %	83.8 %	91.9 %
37/37 operations	37/37 operations	31/37 operations	34/37 operations

Agenda

I. Introduction and Motivation
II. A Closer Look at PaaS
III. Selecting the ‚right' PaaS
IV. Facilitate Portability Efforts for PaaS
V. Related Work
VI. Conclusion

V. Related Work

- Portability for Clouds in general
- PaaS-specific platform issues:
  - PaaS Models, Categories and Ontologies
  - Support for Platform selection
  - Support for Portability
  - Support for Interoperability
- Different Approaches based on
  - Standard(s/proposals)
  - Suitable Abstraction Layers

often combined

Portability by standardization?

- Standards did not help much in other areas, e.g. Process Engines for BPEL or BPMN
- **Industry Vendors are reluctant to accept** …
  - Vendors ‚like' lock-in and ‚special' features
  - 90% standards do not help: missing features
- Lots of approaches focus on IaaS:
  - Open Cloud Computing Interface (OCCI)
  - Cloud Infrastructure Management Interface (CIMI)
  - …
- Standardized descriptions may help for PaaS …

Standard Approaches targeted to PaaS

- OASIS TOSCA:
  - Goals: Portability, Interoperability, Automation
  - TOSCA ‚Language' describes:
    - topologies, relations, properties, dependencies
    - how to orchestrate deployment & management (plans)
  - TOSCA implementations: OpenTosca, …, Cloudify/ARIA
- OASIS **C**loud **A**pplication **M**gmt for **P**latforms:
  - APIs: build, run, …, monitor, patch PaaS applications
  - self-service management: portable ‚across clouds'
  - semantic model based on ontologies
  - implementation status? (Spec V1.2, 05/2017)

Abstraction Layers for PaaS - 1

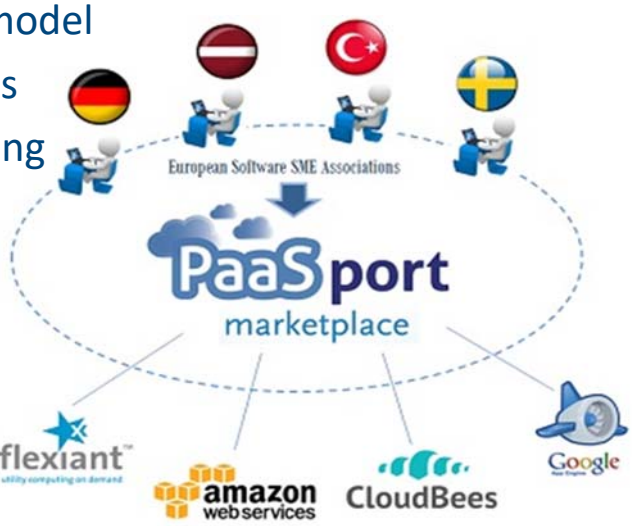
- Cloud4SOA
  - [Andria et al., Service Oriented and Cloud Computing, 2013]
  - General Architecture similar to Nucleus design
  - PaaS Semantic Interoperability Framework (PSIF) model
  - adapts to: AWS Elastic Beanstalk, Cloud Foundry, Openshift, CloudBees, Heroku, cloudControl
- Follow-Up Projects:
  - SeaClouds
    - [Brogi et al., Proc. Conf. Cloud Comp. & Service Science, 2014]
  - CloudPier
  - ?Unicorn?

Abstraction Layers for PaaS - 2

- PaaSManager:
  - [Cunha et al., CSIS 11(4), 2014]
  - adapts to: CloudBees, Cloud Foundry, Heroku
- **C**ompatible **O**ne **A**pplication & **P**latform **S**ervice
  - [Sellami et al., Proc. Conf. Cloud Computing 2013]
  - adapts to: Cloud Foundry, Openshift, Google App Engine
- Cloud Foundry Core Definition:
  - common capabilities for runtime & native services
  - API for checking the state of a platform w.r.t. capabilities
- OW2 FraSCAti: OASIS SCA-based interface to Clouds
- …

Models to describe PaaS - 1

- PaaSPort-EU: ‚Market Place'
  - Ontology-based semantic PaaS model (CAMP-OWL)
  - relies on results from Cloud4SOA
  - re-uses the CAMP semantic model combined with vendor studies
  - Elaborate matching and ranking based on functional and non-functional criteria (hierarchical)



Models to describe PaaS - 2

- mOSAIC: Open-Source API Platform for Multiple Clouds
  - [Moscato et al., IEEE FedCSIS 2011]
  - Ontology for ‘all’ aspects of cloud offerings?
  - very general high-level categories for all cloud models (IaaS)?
  - Abstraction layer: ‘common API to use Cloud services’
- MODAClouds: model-driven approach using DSLs
  - [Di Nitto et al., Model-Driven Development & Operation of Multi-Cloud Applications, Springer Briefs in Applied science, 2017, OpenAccess]
  - ‘comparison engine’ compares deployment models
  - ‘deployment & resource provisioning engine’ based in VMs
- ...

VI. Conclusions

- What is on offer:
  - Model for PaaS offerings
  - Catalog of >70 vendors
  - Easy selection and matching through dashboard
  - Nucleus API for porting management aspects
- Current & Future Work:
  - Partial ‚Semantic‘ Matching Algorithm
  - Real-world scenario with a start-up company

Published Papers on PaaS Topics

- Kolb, S., Wirtz, G.: **Data Governance and Semantic Recommendation Algorithms for Cloud Platform Selection** Proceedings of the 10th IEEE Conference on Cloud Computing, Honolulu, HI, USA, June 25 - June 30, 2017.
- Kolb, S., Röck, C.: **Unified Cloud Application Management**
  Proceedings of the 12th IEEE World Congress on Services, San Francisco, USA, June 27 - July 2, 2016.
- Röck, C., Kolb, S.: **Nucleus - Unified Deployment and Management for Platform as a Service**
  Bamberger Beiträge zur Wirtschaftsinformatik und Angewandten Informatik Nr. 100, University of Bamberg, April 2016.
- Kolb, S., Lenhard J., Wirtz G.: **Application Migration Effort in the Cloud**
  Services Transactions on Cloud Computing (STCC), 3(4), 2015.
- Kolb, S., Lenhard J., Wirtz G.: **Application Migration Effort in the Cloud - The Case of Cloud Platforms**
  Proceedings of the 8th IEEE International Conference on Cloud Computing (CLOUD), New York, NY, USA, June 27 - July 2, 2015.
- Kolb, S., Wirtz G.: **Towards Application Portability in Platform as a Service**
  Proceedings of the 8th IEEE International Symposium on Service-Oriented System Engineering (SOSE), Oxford, United Kingdom, April 7 - 10, 2014.

Thank you!

Questions?!

Stefan Kolb & Guido Wirtz

@stfnklb

https://PaaSfinder.org

{stefan.kolb|guido.wirtz}@uni-bamberg.de