FluidCloud

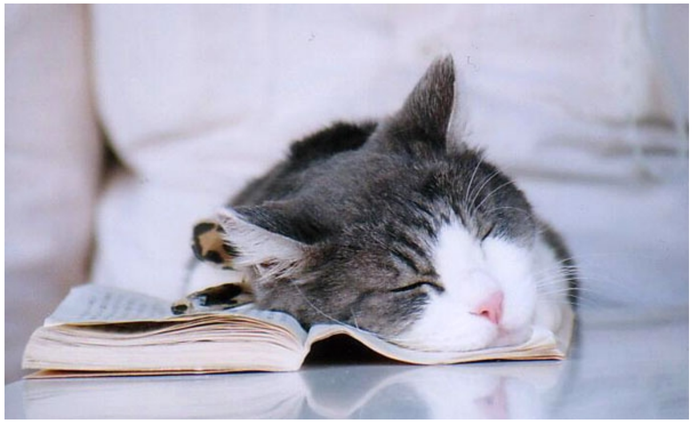
An Open Framework for Cloud Service Relocation

hello USENIX!

- I'm Andy Edmonds (@dizz)
  - Zurich University for Applied Sciences
- Thanks to my co-authors
  - Thijs Metsch (Intel)
  - Dana Petcu (IeAT)
  - Erik Elmroth (Umea University)
  - Jamie Marshall (Prologue)
  - Plamen Ganchosov (CloudSigma)

tl;dr

- enables relocation of service instances
- integrated framework & approach
- extensible
- can solve interoperability
- rich driver of research



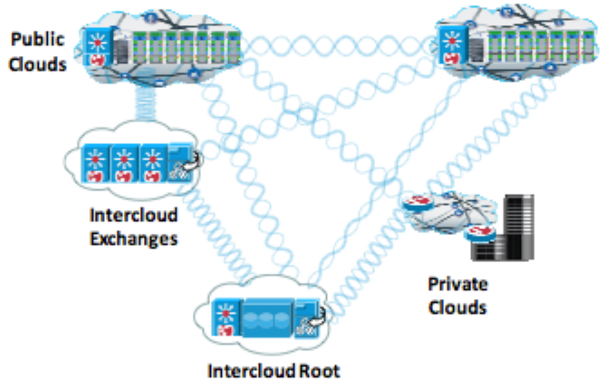
now onwards.... or to sleep!

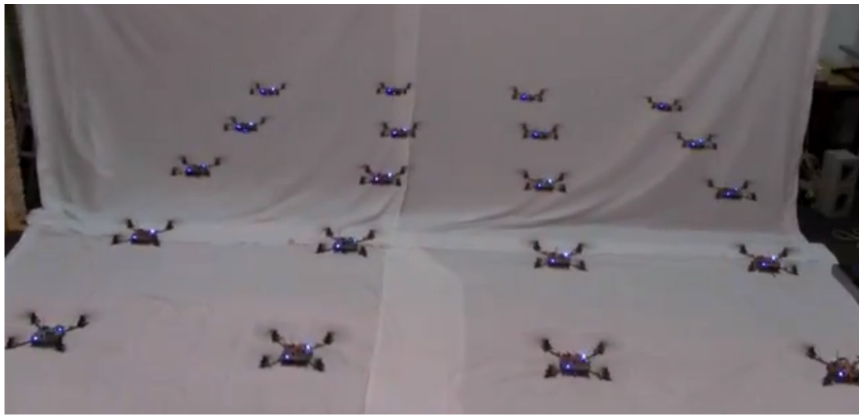
overview

- history
- key question
- why?
- for what?
- the enablers
- architecture
- implementation
- results
- future

FluidCloud…

- where did it come from?
  - an email migration tool
    - yippiemove.com
  - a future cloud vision
    - InterCloud
  - a swarm of quadcopters
    - dynamic adaptation to changing environment





www.cloudcomp.ch

key question

"How to intrinsically enable and fully **automate relocation** of service instances between clouds?"

key question

“How to intrinsically enable and fully **automate relocation** of service instances between clouds?”

FluidCloud

www.cloudcomp.ch

but why?

- other than a metaphor…
- ease of movement
  - just like water flowing
- provider independence
  - reduce risk
  - provide “insurance”

… disparate techs

Standards	Tools/Libraries	Misc
Occi Open Cloud Computing Interface	δ·CLOUD	Cloudant
IETF®	libcloud	globus online™
SNIA	jclouds™	
DMTF®		

all cool…

- **BUT** -

not integrated,

features missing

a need?

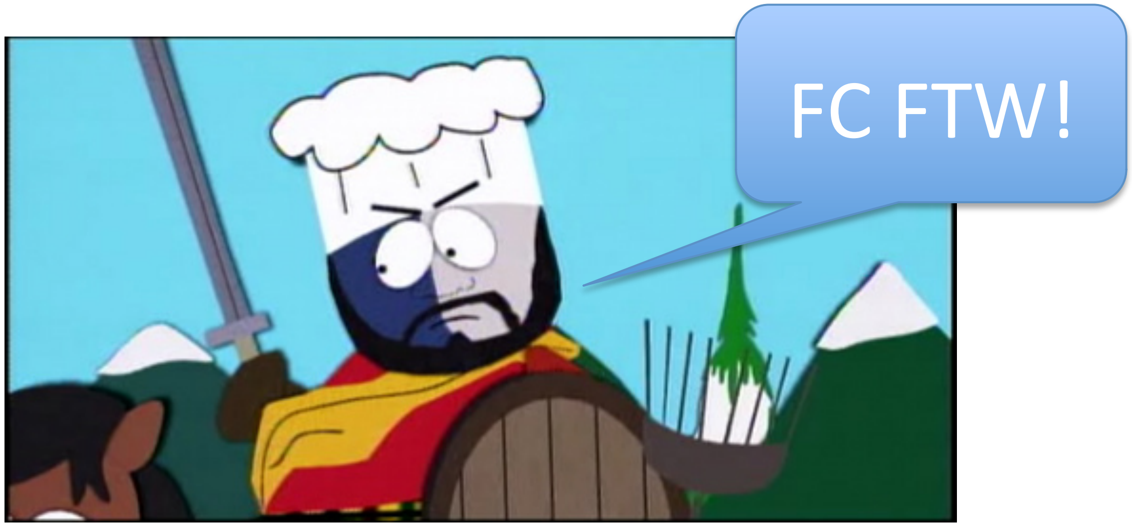
# of entry points & partial solutions
increased complexity of integration

Framework for Service Relocation

ultimately...

remove lockin

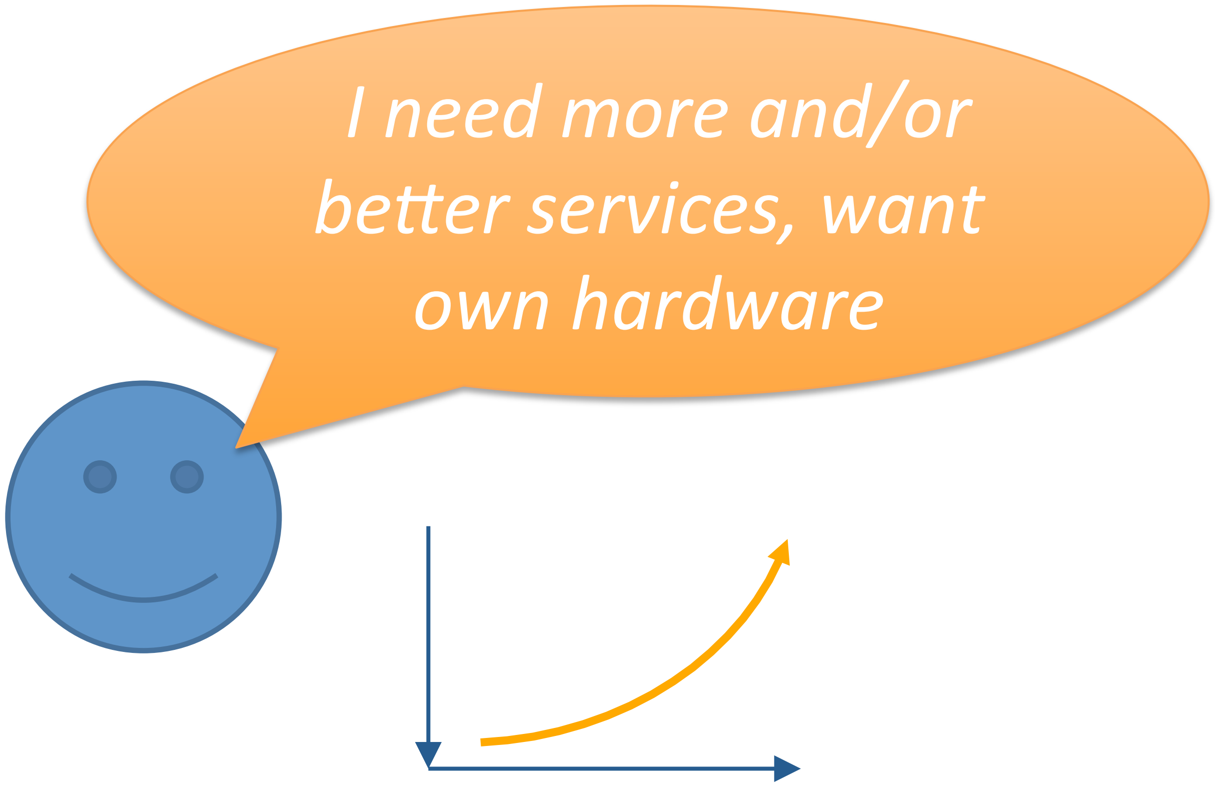
free your services



but for what?

- considered from two points of view
  - business-oriented use cases
  - technical use cases

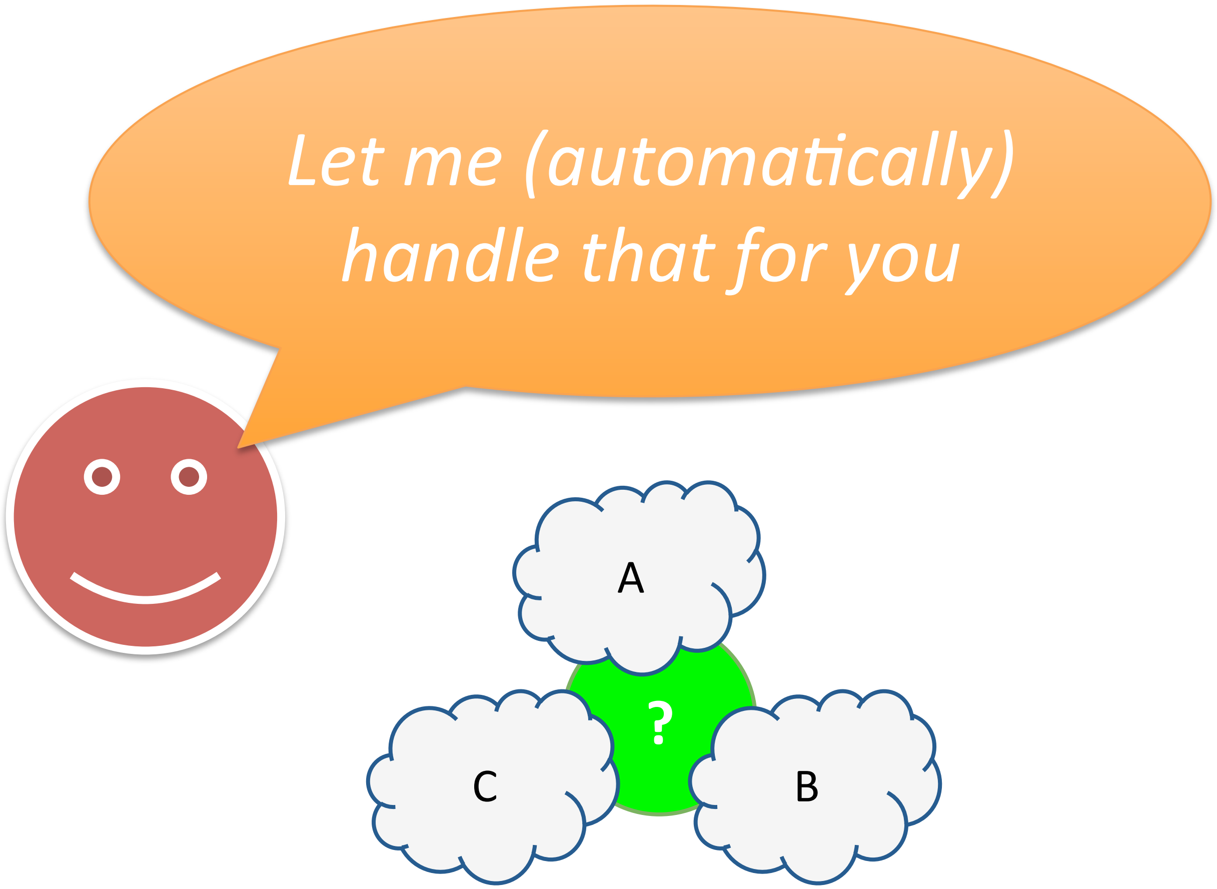
biz use case 1
“The Startup”



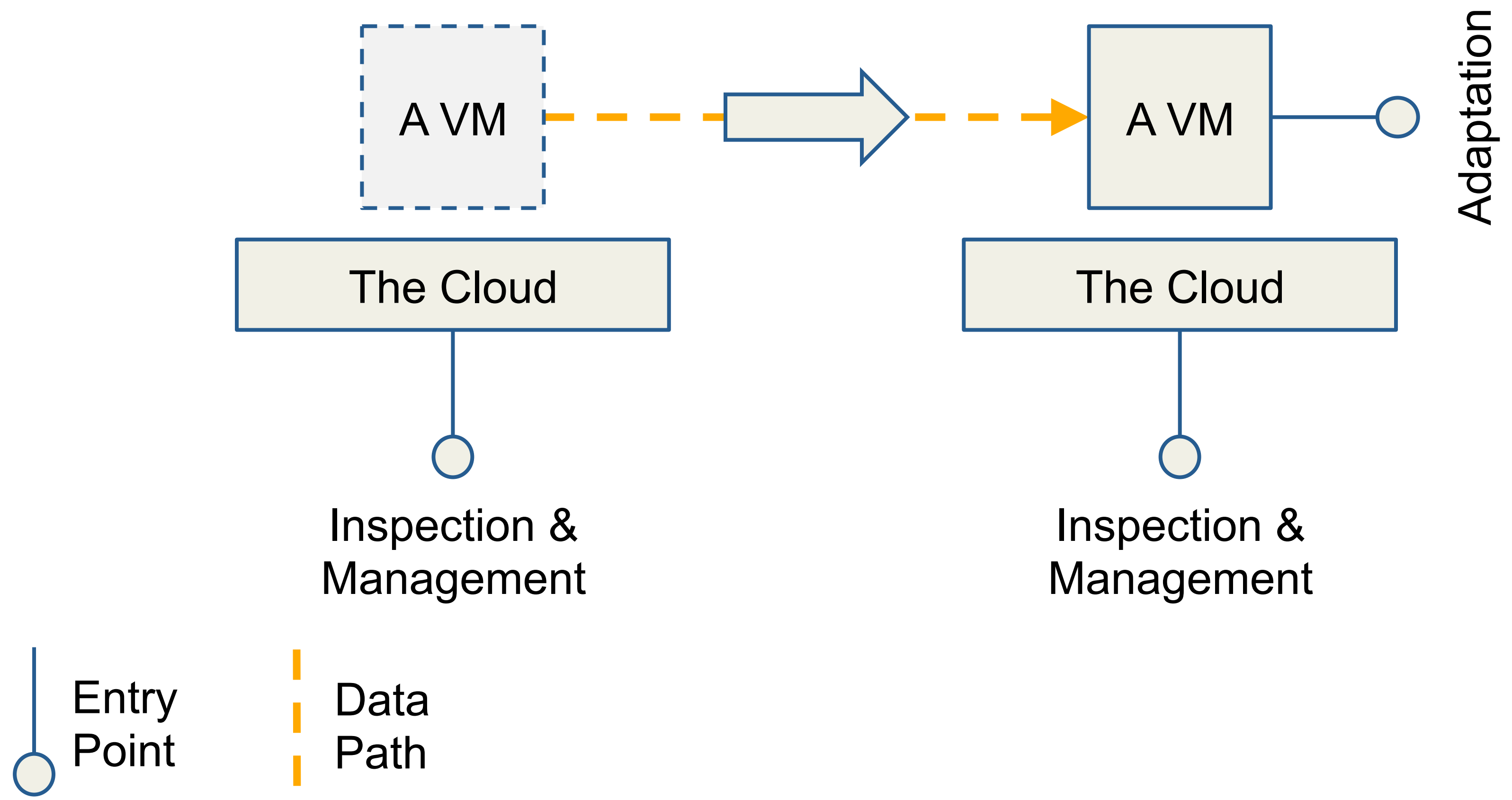
biz use case 2
“The Cloud Service Provider”

Happiest Darth Vader, evar!

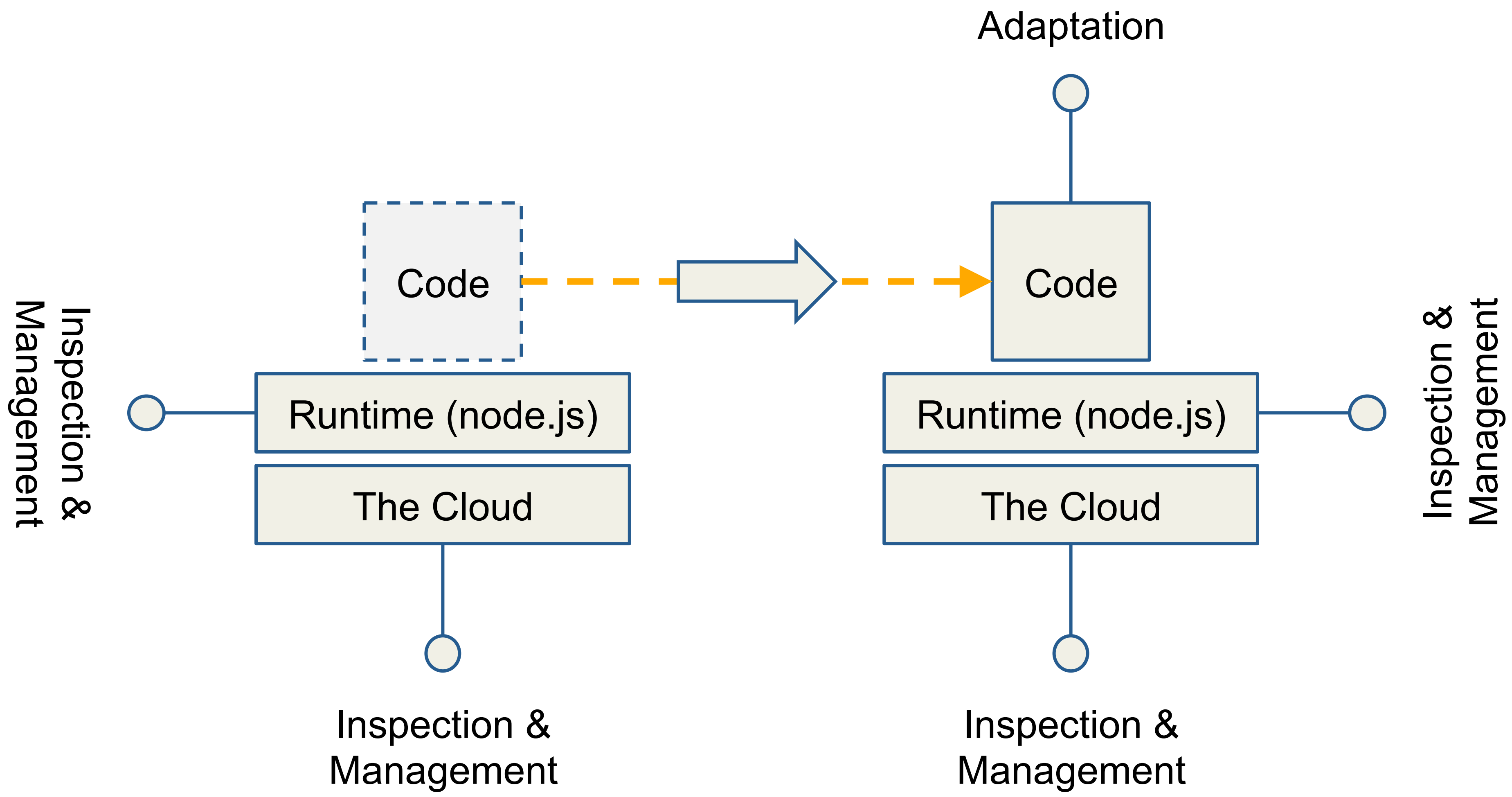
biz use case 3
"The Cloud Broker"



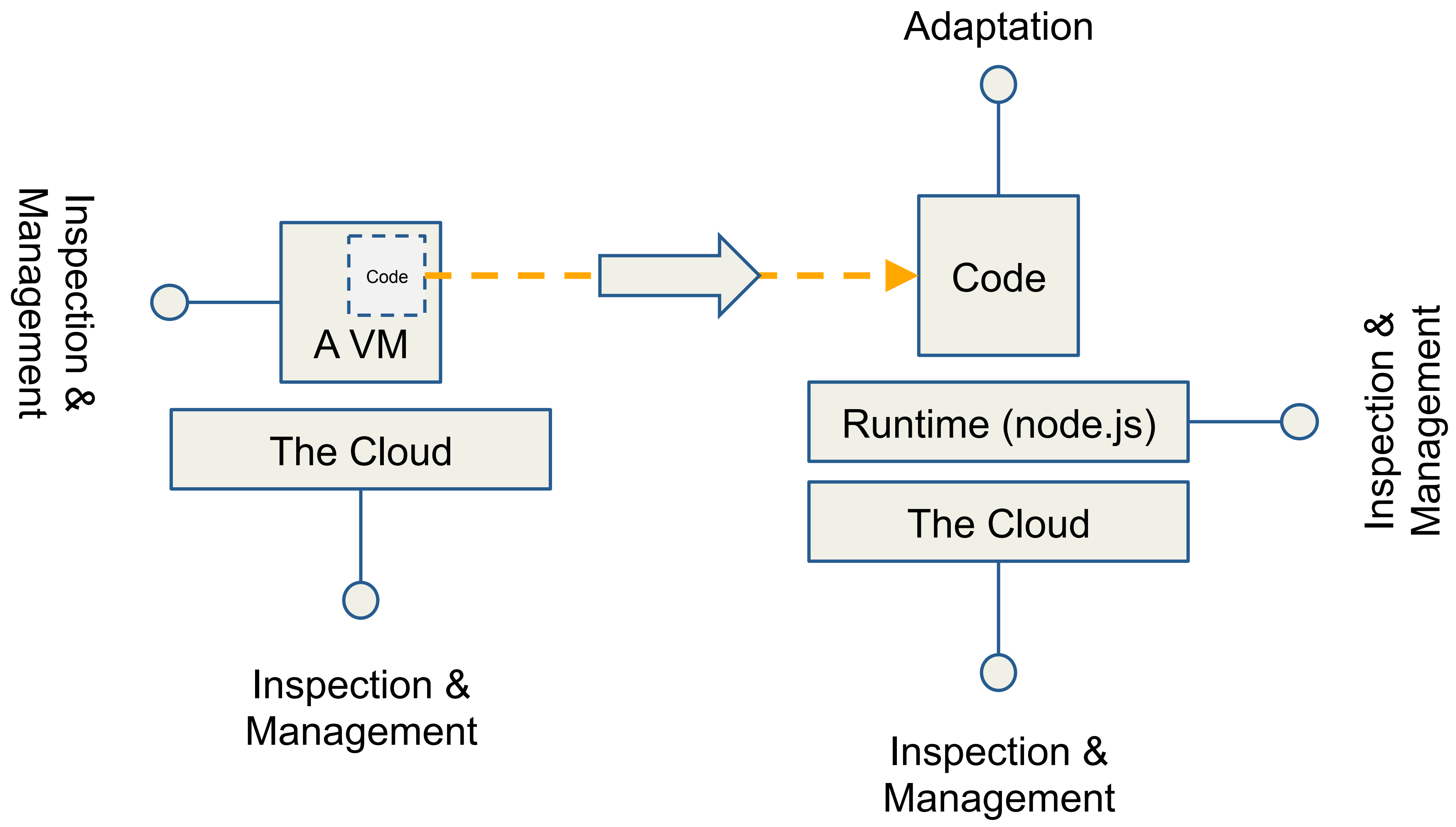
tech use case 1
relocating IaaS



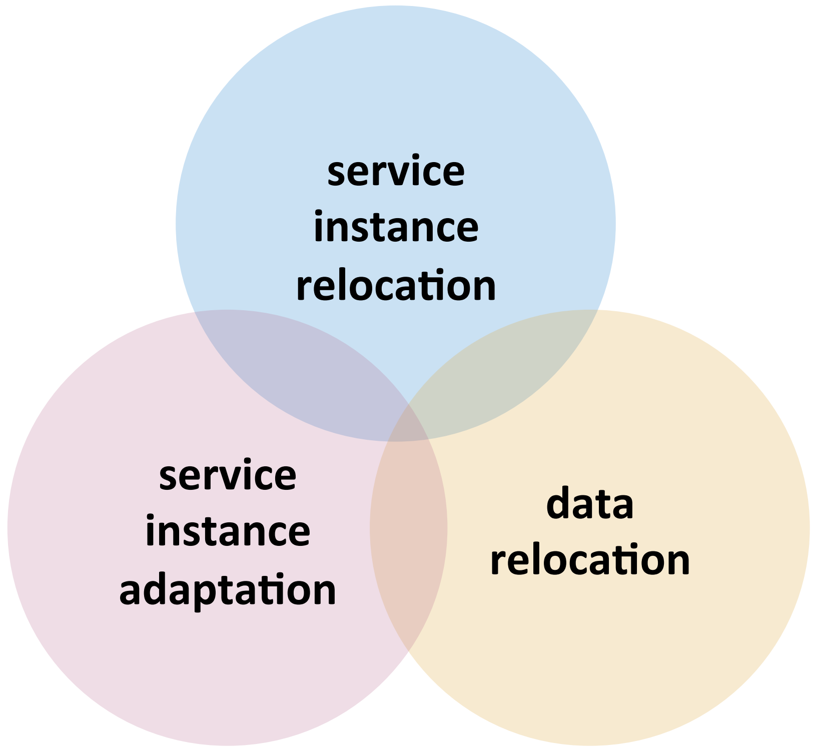
tech use case 2
relocating PaaS

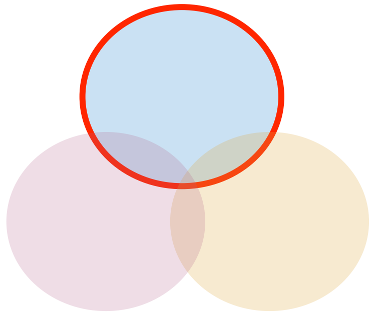


tech use case 3
(possibly) IaaS to PaaS



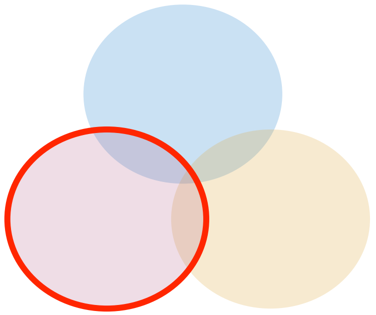
key enabling concepts





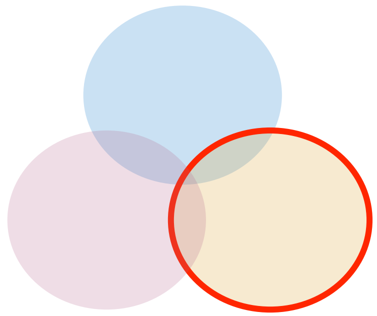
service instance relocation

ensuring the overall **orchestration** and **process of moving** a cloud service from the source to the target cloud service provider.



service instance adaptation

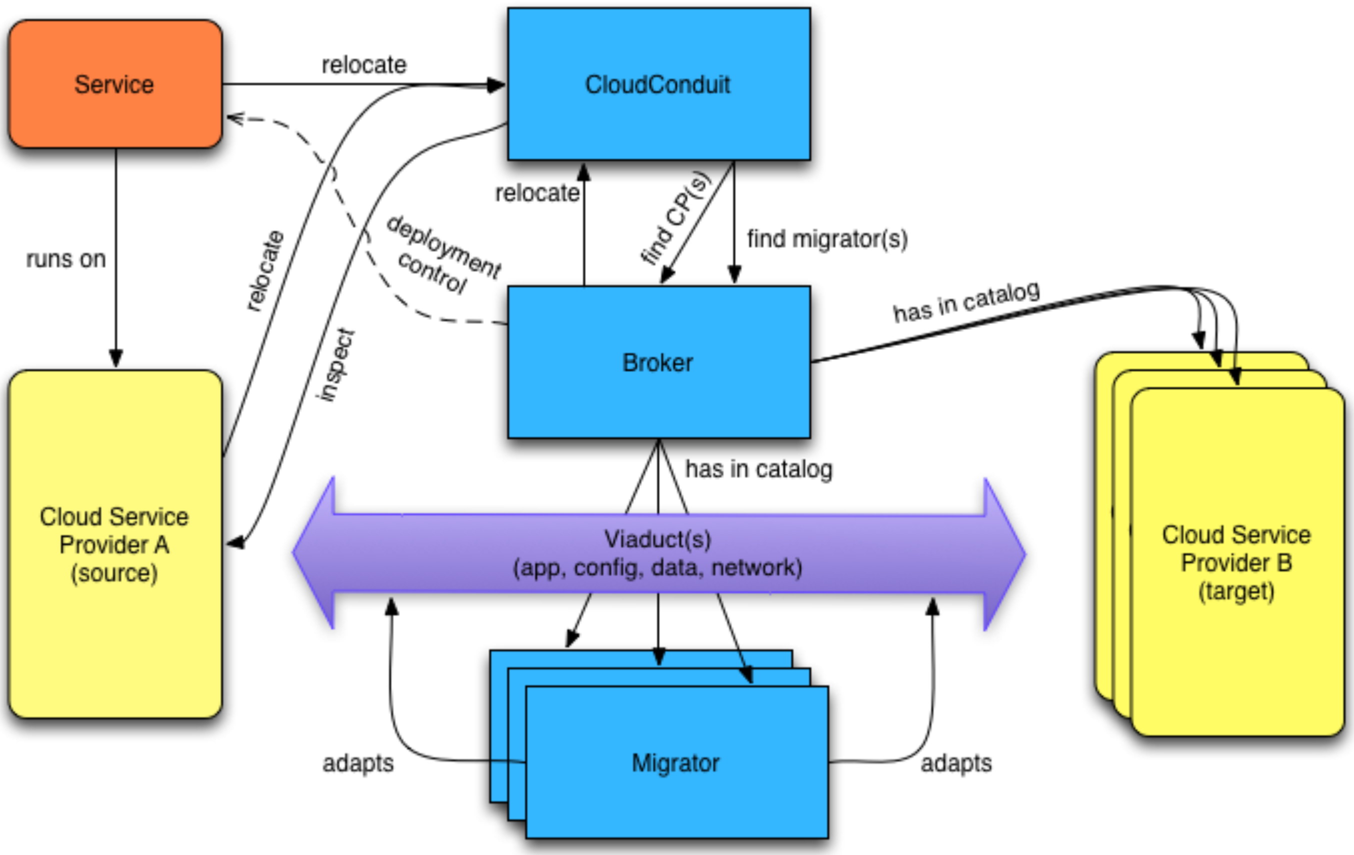
conversion, **adaptation**, transformation and movement **of the service** and its related data.

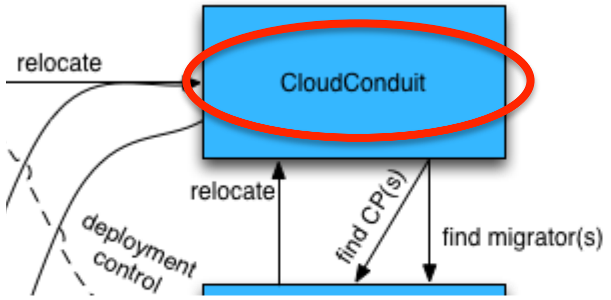


data relocation

relocation, migration, transformation and conversion of the **data belonging to the service**.

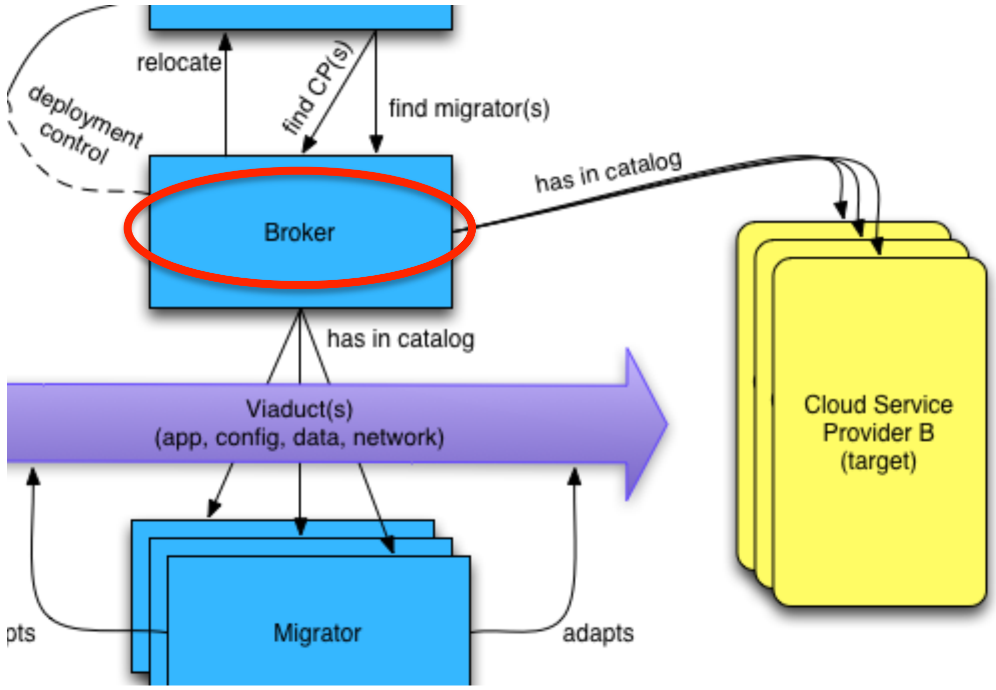
architecture





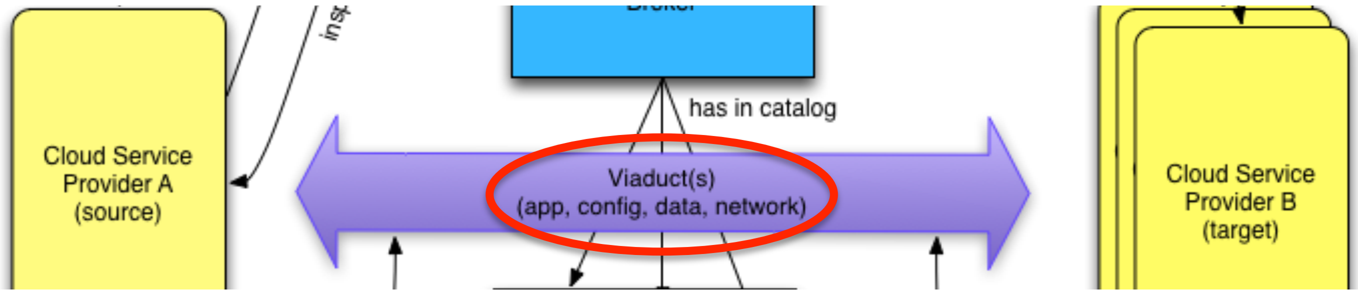
Cloud Conduit

- **orchestrates** the process,
- **introspects** the service instances (incl. topology) to be relocated.



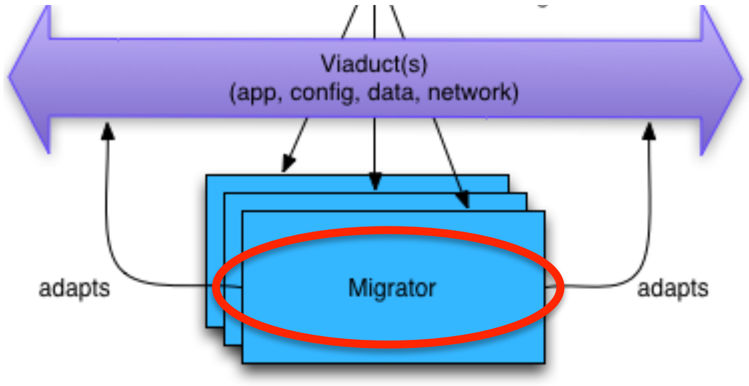
Broker

- discovers, matches and provides both cloud provider services and ***Migrator*** facilities



Viaduct

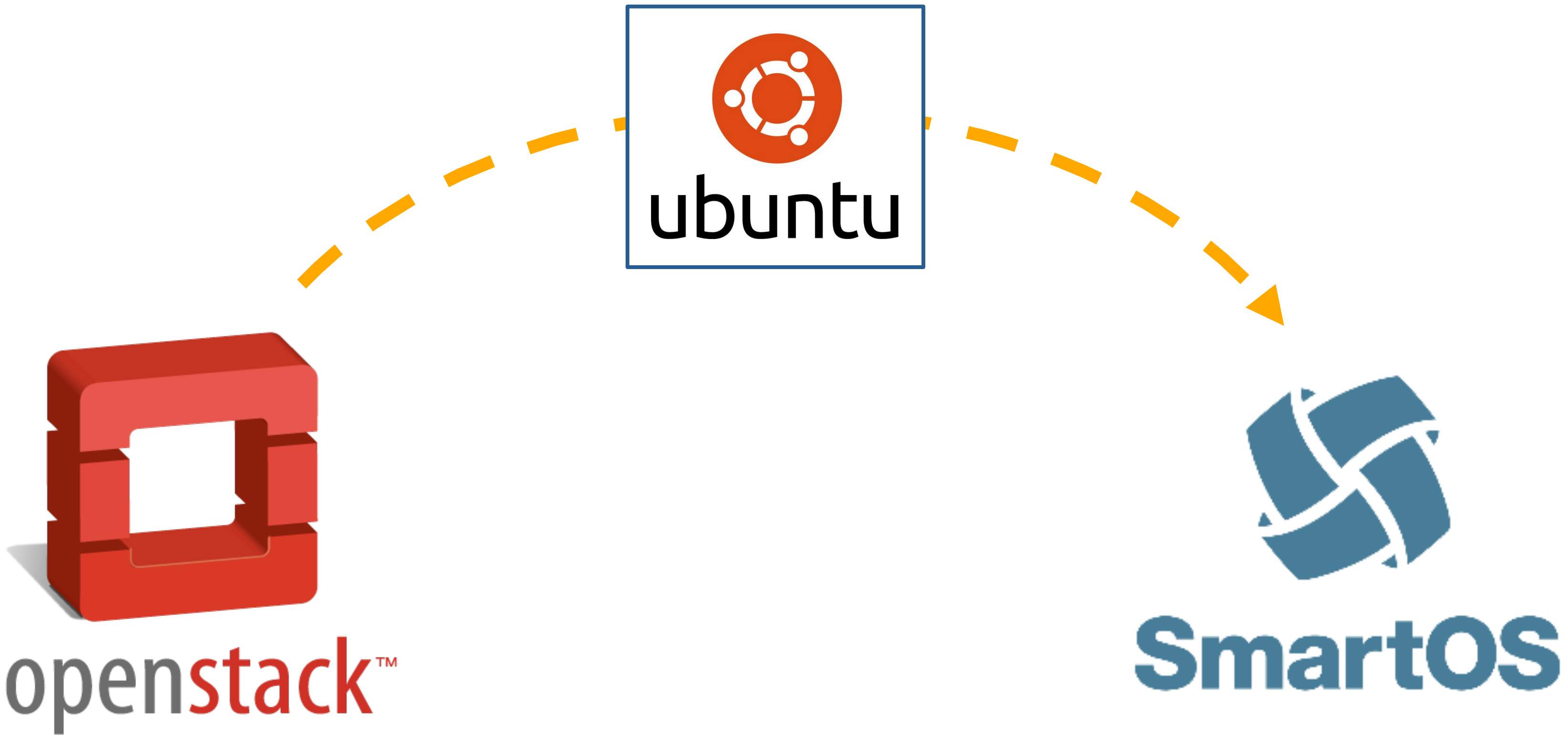
- **logical path** between two providers in which Migrators are organised (as workflow)



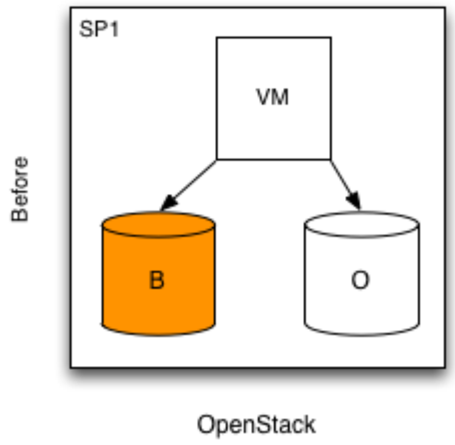
Migrator

- libraries, tools and services for adaptation
- **one specific task related to relocation** (possibly partial) of service instances

POC implementation scenario



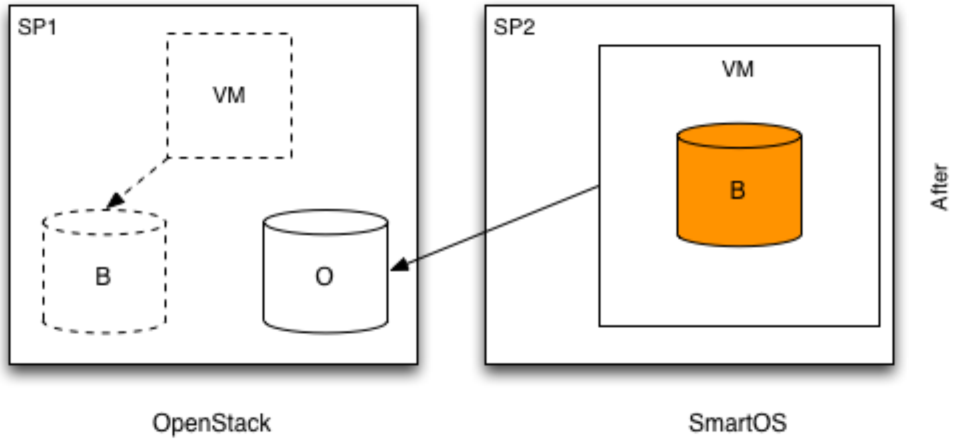
POC implementation scenario



before: all services in one provider

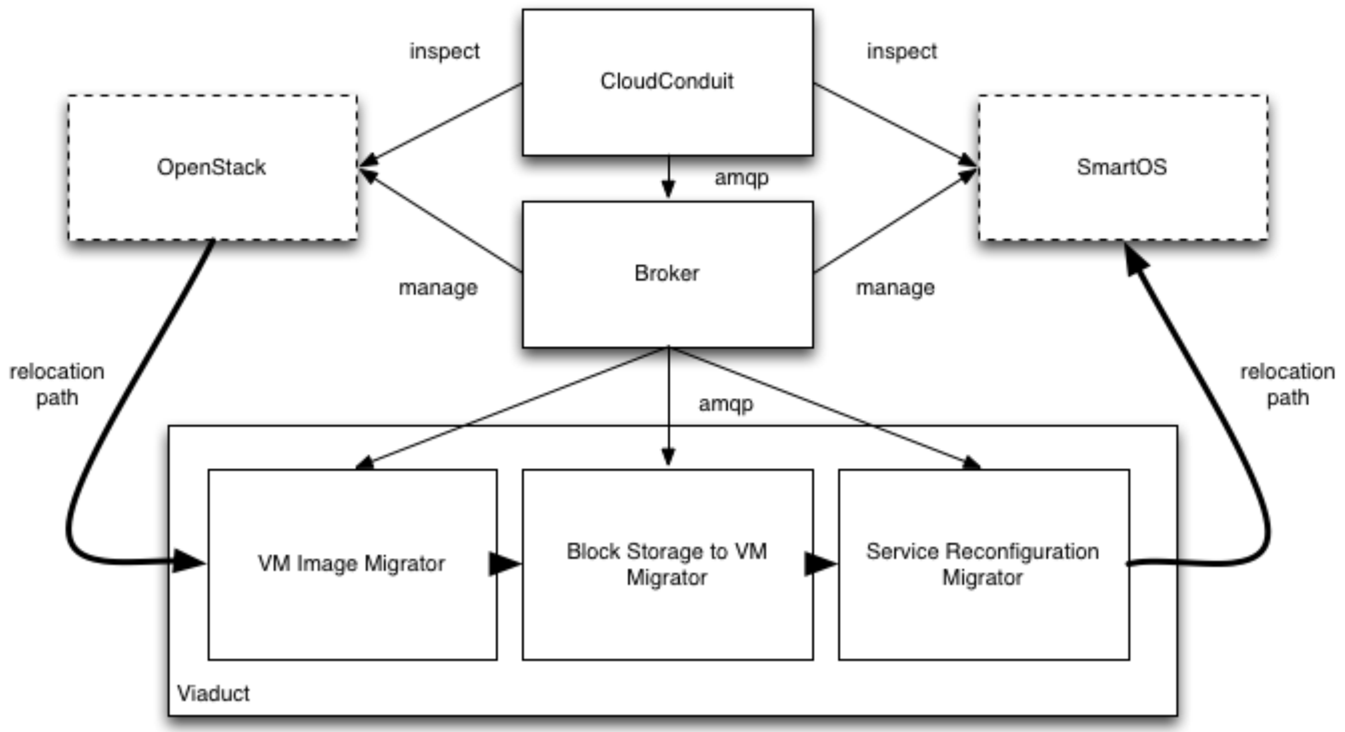
Adaptation of:
- Service configuration
- Block Storage
- Access to Object Storage

after: all but object storage moved to new provider



www.cloudcomp.ch

POC implementation



python

www.cloudcomp.ch

results

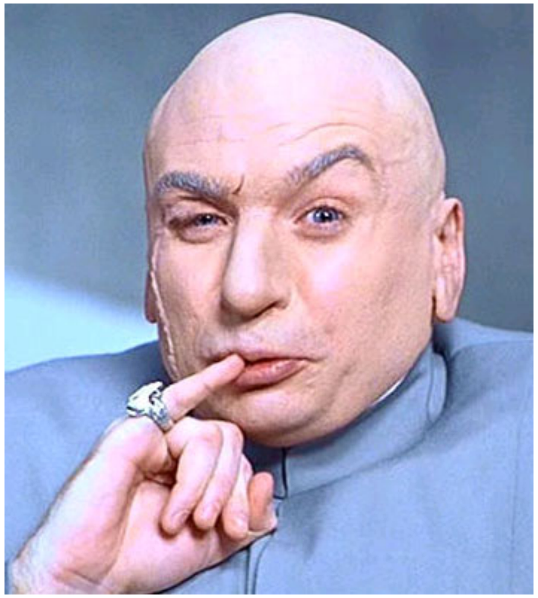
- successful! Architecture is appropriate… so far :-)
- initial metrics
  - 1GB switched network,
  - POC relocation accomplished in approx. 10 mins
    - ~5 for VM (5.4GB)
    - ~1 for 512 test file (on block storage)
    - ~10 secs for reconfig
  - data transfer is the time-heavy component
    - How long will Tanenbaum's station wagon remain?
- *note:* perf. metrics were not the goal

is it crazy?

- similar things, some in diff domains
  - pi.pe
  - CloudVelocity
  - Racemi CloudPath

further work

- service decomposition over multiple providers
- investigate IaaS to PaaS
- more on adaptation and inspection
- data payload optimisations
- work on PaaS to PaaS



thanks!

questions?

no? i have some for you!

www.cloudcomp.ch

questions

- do providers really want lock-in?
- is such a thing needed or is it niche?
- are the use cases realistic?
- what do you see as infeasible?
- are there simpler approaches?
- do you know of similar things?
- was something unclear?
- was something missing?