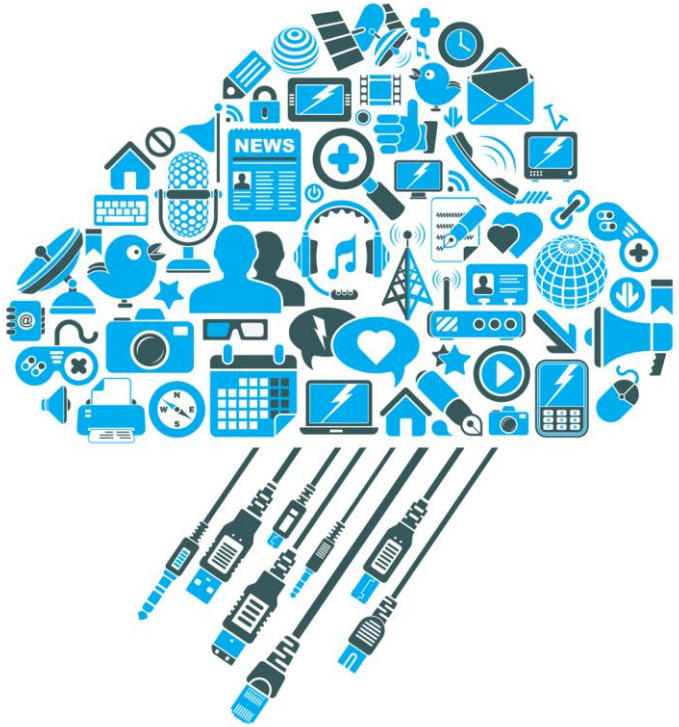
Sky&Sand

CLOUD COMPUTING

THOMAS BOLTZE
CTO SKY & SAND
THOMAS.BOLTZE@SKY-SAND.COM

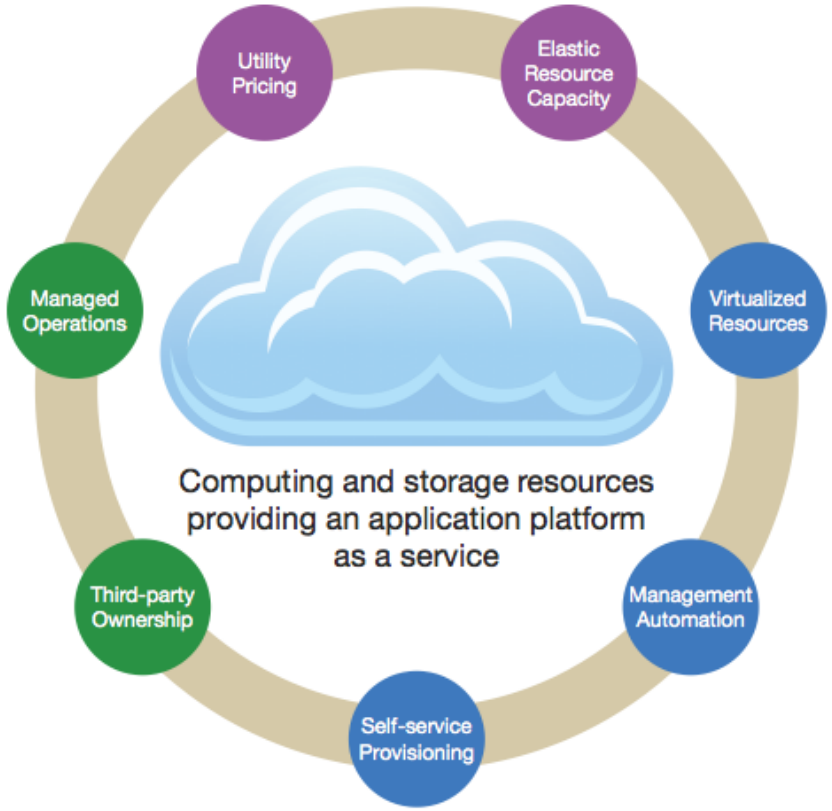


What is Cloud Computing?

- A new way to bill for resources:
  - Hourly (virtual server, RDS)
  - Data volume (S3, Glacier, SendGrid)
  - Transactional (email, SMS, credit card)
- Major shift of TCO
  - Lower initial & fixed cost
  - Scaling of variable cost
  - But pay you must
- Hype – we have been using most of this for a long time
  - Virtual Servers, Services, APIs,

Elements of Cloud Computing

Concept: MWD Advisors — www.mwdadvisors.com



Economic Elements:
Pay-as-you-go, pay-as-you-grow, no CAPEX.

Architectural Elements:
Simple, abstract environment for development.

Strategic Elements:
Focus on your core business, leave the rest to someone else.

Pros & Cons

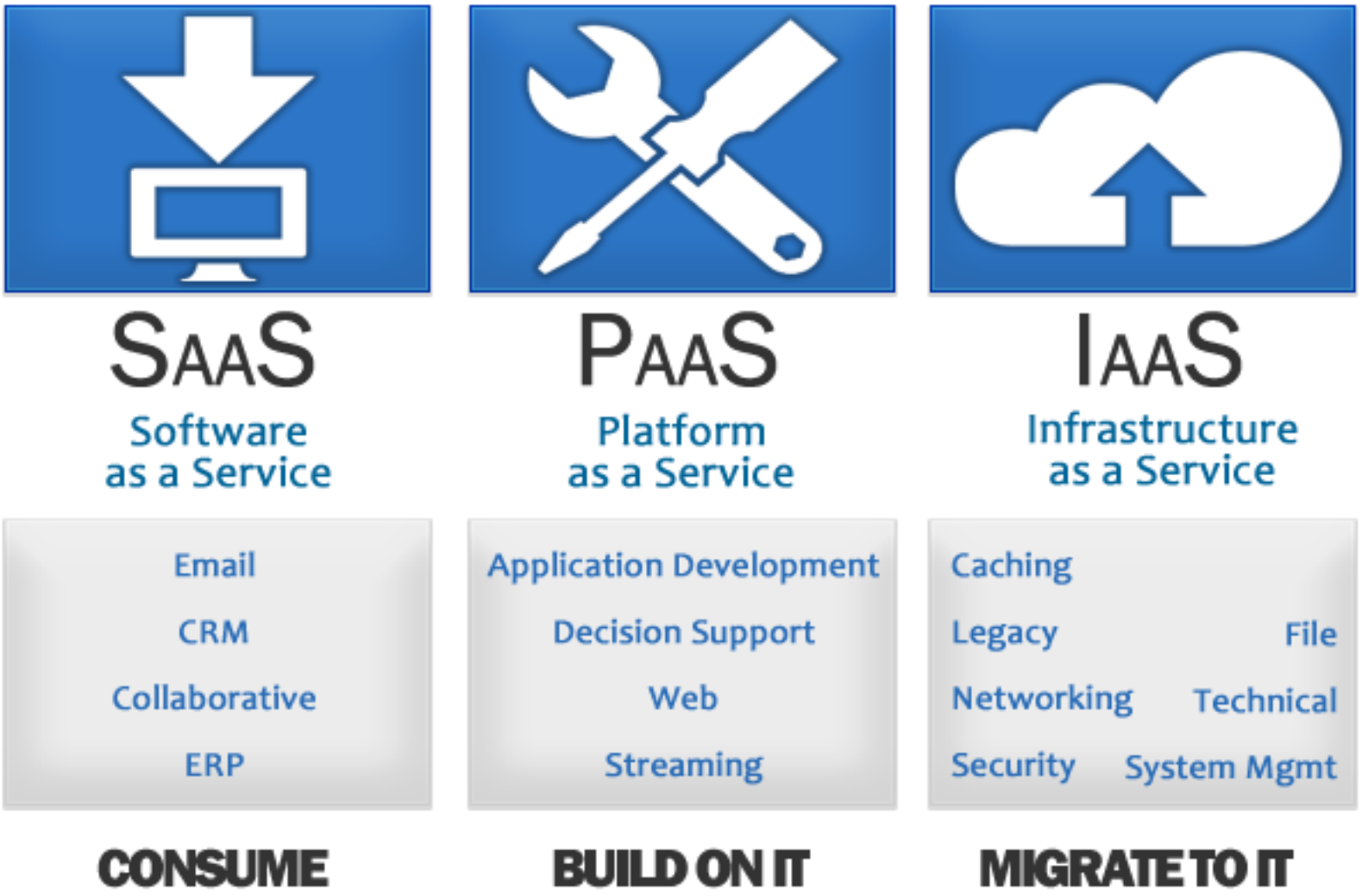
- Pros
  + Ease of use
  + Low startup cost
  + S.E.P.
  + Continuous evolution
  + Focus on core business
- Cons
  – Lock-in
  – Lack of control
  – Cost at scale
  – Security is not in your hands (can be a plus too)

There is no right solution – but **probably** cloud is the right choice

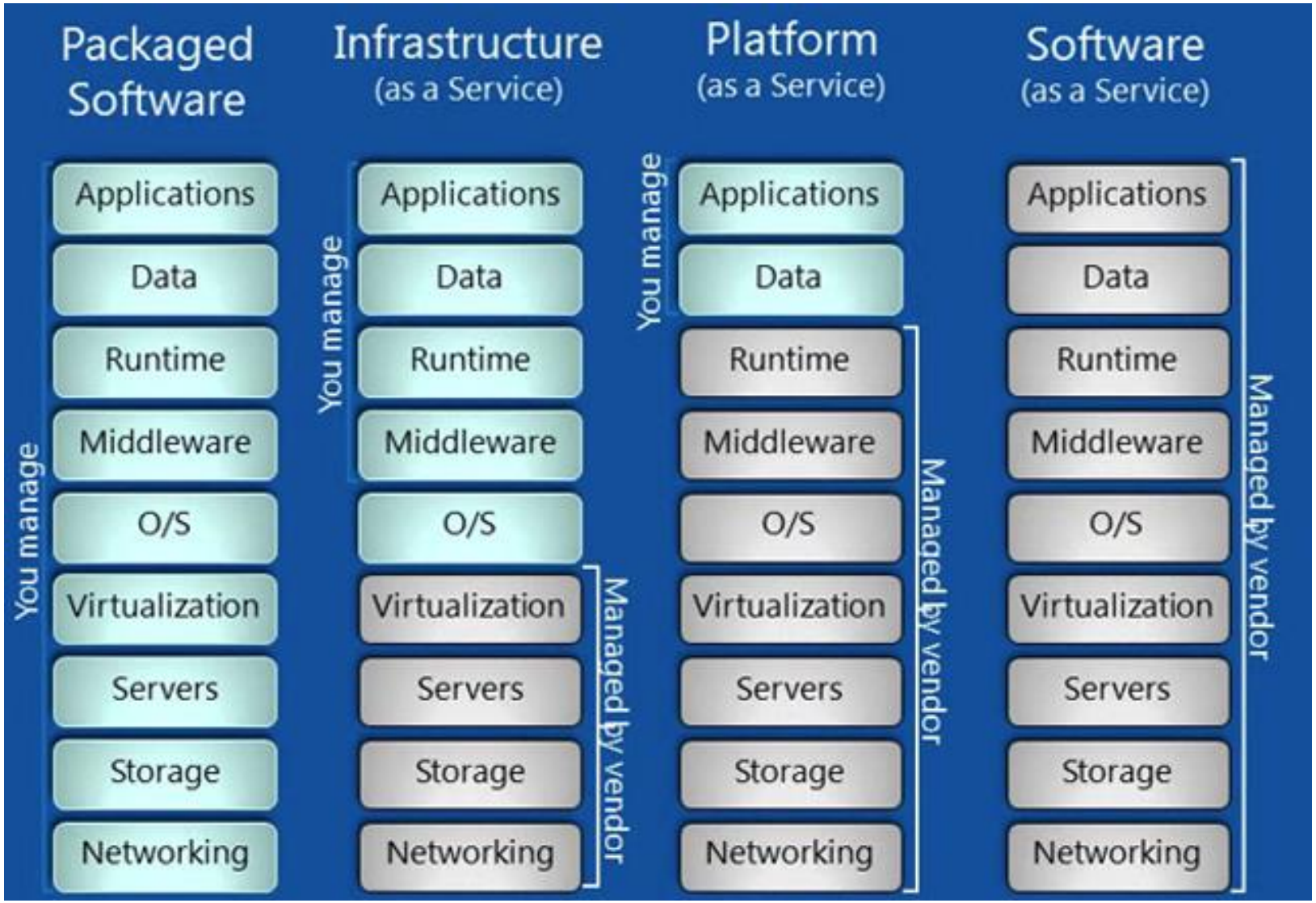
IaaS, PaaS, SaaS

- Infrastructure as a Service
  - Rackspace, Amazon, Google Compute Engine, Microsoft Azure
  - Servers, Storage, Firewalls, Loadbalancers (Virtualization)
- Platform as a Service
  - Heroku, EngineYard, Google AppEngine, AWS Elastic Beanstalk
  - Application Stack Virutalization: Workers, Dynos
- Software as a Service
  - Amazon RDS, Google Mail, Stripe, SendGrid, SalesForce CRM
  - Integrate external services via API‘s

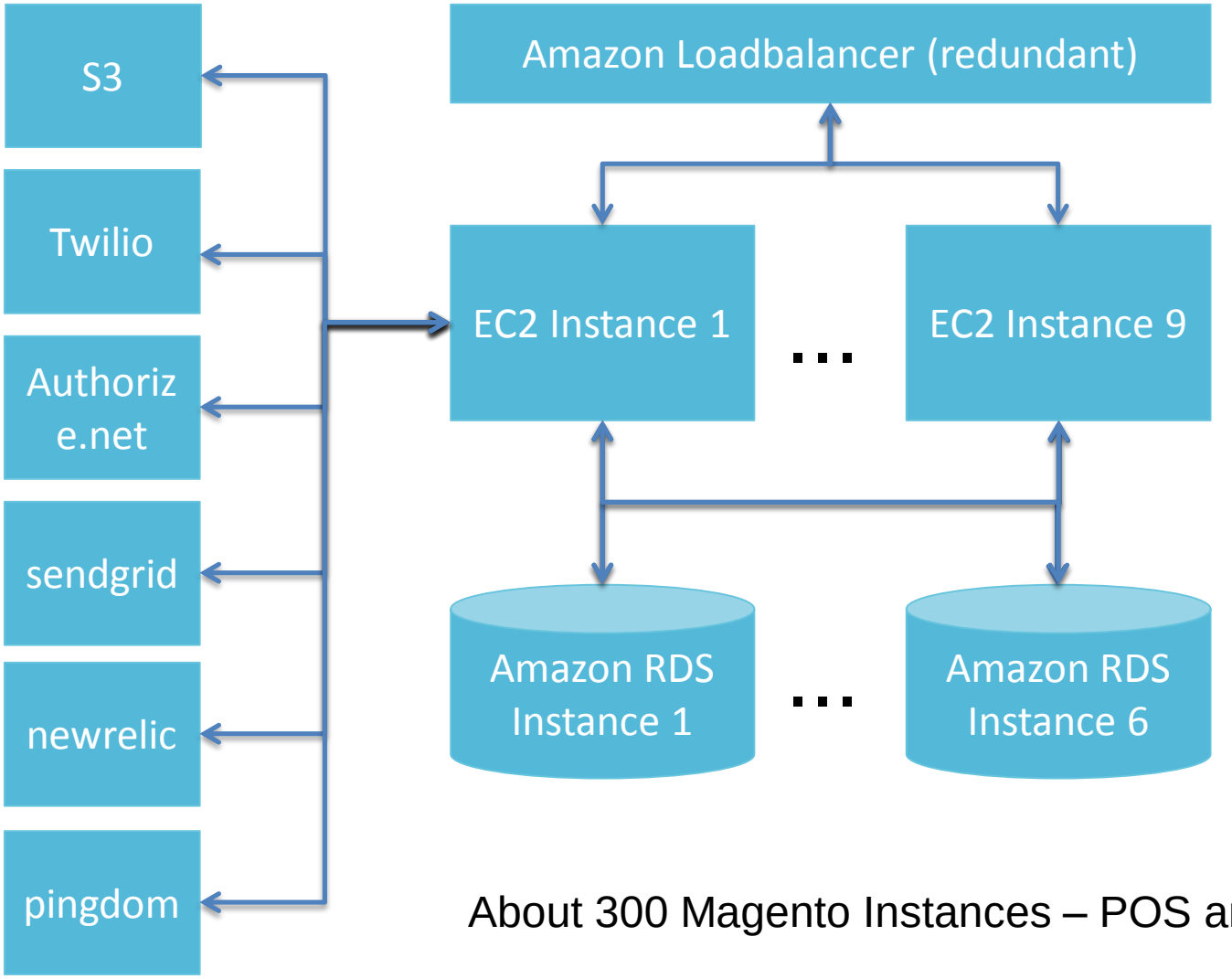
Flavours of Cloud Computing



What you need to master



Real-world example: floranext.com



Multi-AZ
No sharding
Memcached
Apache + fastCGI
Rails + unicorn
Gluster
Monit

Multi-AZ
Sharding by Customer

About 300 Magento Instances – POS and Websites

Floranext.com History

- Single Rackspace Server for MySQL + Single Rackspace server for Apache & PHP (osCommerce + shell scripts)
- Multiple Rackspace web servers
- Migration osCommerce to Magento (performance nightmare)
- Bigger DB server, more web servers
- Meltdown during server resizing
- Overnight move to Amazon
- Single RDS instance, + 4x EC2 instances
- Massive performance hit
- Continuous changes
  - Up to seven RDS instances, 14 EC2 instances
  - Different sizes, different configs
  - Valentine‘s day, Mother‘s day

Total Cost of Ownership

- Colocation / Own Servers
  - Fixed cost for servers and colocation / hosting is high
  - Initial costs are high
  - Cost of change is high
  - Flexibility is low
  - Waste is high

- IaaS, PaaS & SaaS
  - Initial costs very low, mostly training & learning
  - Hourly / usage based rates grow fast (look out for those Amazon bills)
  - The higher the level of abstraction the higher the cost er page view.

Pitfalls

- Redundancy, Scaling, Backups
  - Still need to worry about these
  - ☺ Testing is cheap
- Not all servers are created equal
  - Rackspace vs Amazon: 2x power / buck (old benchmark)
  - Performance is still expensive
  - Use reserved instances
- Migration can be a headache
  - E.g. from single RDS instance to six instances
  - From Rackspace to Amazon
  - Automate everything

Colo vs IaaS vs PaaS

- PaaS at the beginning of a project
  - E.g. Heroku
  - Littleunderstanding of Servers and hosting required
  - Low $$ committment
  - Git push and your done
  - Gets expensive very quickly
- Move to IaaS for scaling
  - Better $$ / pageview
  - More control
  - Sysadmin required
- Colo for massive scaling
  - Best cost structure for large scale
  - Requires Sysadmin(s)
  - Augment with IaaS for plannable peaks

When to use SaaS

- Always, unless:
  - This is the core product of your company
  - It becomes too expensive (Mongo, Emails, Billing)
- Scale your provider
  - MailChimp is cute, SendGrid is serious, and Emarsys is massive
  - Rather switch provider than build it yourself at scale
- Decide on a case by case basis
  - Do I have the right people?
  - Will I really save money?
  - Does it distract me from my core product?
  - Is this a differentiator?

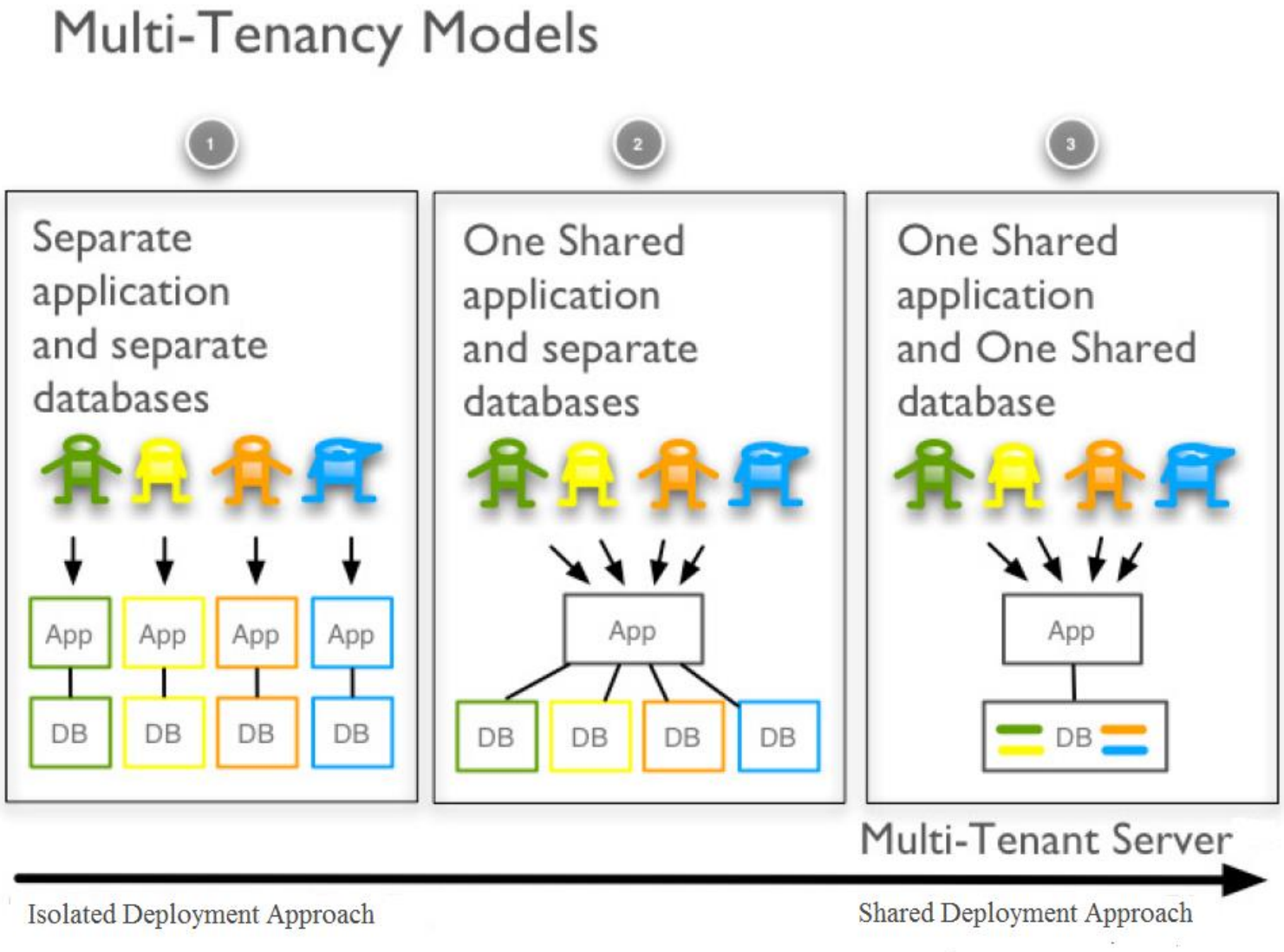
What about my Data?

- Legal Compliance
  - HIPAA for Medial Records in the US
  - European Data Protection Regulation
  - Datenschutz in DE
  - Check each country
- PCI Compliance
  - Credit Card numbers
  - Physical access to servers
  - Logical access to data
- Security
  - Can you build better (physical) protection for the data than a hosting provider?
  - Have you tried hacking into your site?
  - Passwords, SSH keys, phpMyAdmin
  - Should all your developers have access to production data & systems?

What About OpenStack?

- OpenStack is an open source version of IaaS tools. (Rackspace & NASA)
- Great for learning about how stuff works
- Great if # of servers > 100

Multi-Tenancy



Multi-Tenancy – Now What?

- Single Instance versus dedicated instances
  - No right answer
    - Dedicated Magento instances on shared RDS/EC2 instances + Single Rails Instance for managing app pool and instances
    - Dedicated ASP.net instances + single Rails instance for API and mobile CMS (with Octopus)
    - Single Rails Instance, everything scoped to current user.
  - Depends on the platform and legal requirements
    - Rails: Very easy
    - Magento: painful
- Scaling Strategies
  - Sharding, Swimlanes, Caching, Caching

Scaling Strategies

- Bigger server(s) – Scale-Up
  - Reaches the limits
  - Not sustainable
  - Redundancy is very expensive: 50k USD DB server + hot stand-by = 100% overhead
  - Design for non-failure
- More server(s) – Scale-Out
  - Application needs to be designed for this
  - Eventually requires sharding
  - Better for cost
  - 3x10k DB server + 1 hot stand-by = 30% overhead
  - Swimlanes
  - Break app into loosely coupled services, e.g. image rendering, pdf creation
  - Asynchronous
  - Design for failure

Monitoring & Performance

- NewRelic.com
  - Rails (and PHP) full stack monitoring and performance analysis
  - Spot core performance bottlenecks – fix – rinse & repeat
- Hourly vs Continous
  - Use professional for a few days when you have problems to fix
  - Free version otherwise ☺
- But I have twenty servers, this is expensive!
  - They all run the same stack …

My Favorite Services

- SendGrid (emails & newsletters)
- NewRelic (performance analysis & monitoring)
- TaxCloud (tax caclcuation US)
- Authorize.net (payment)
- Rackspace (oomph / $)
- Heroku (simplicity)
- Google Analytics
- LoadImpact (load testing)
- Any CDN (faster is faster)
- BrowserStack (real browser testing)
- github

Try before You Buy

1. Sounds nice on paper
2. Prototype it
3. Benchmark it (at scale)
4. Try to break it (if you succeed try to fix it)
5. Design & build redundant solution (as robust as you need)
6. Really understand the cost (if it matters)
7. Launch it

Thank You!