The Evolution of PaaS

QCon London 2012

Paul Fremantle
CTO, WSO2
paul@wso2.com
@pzfreo

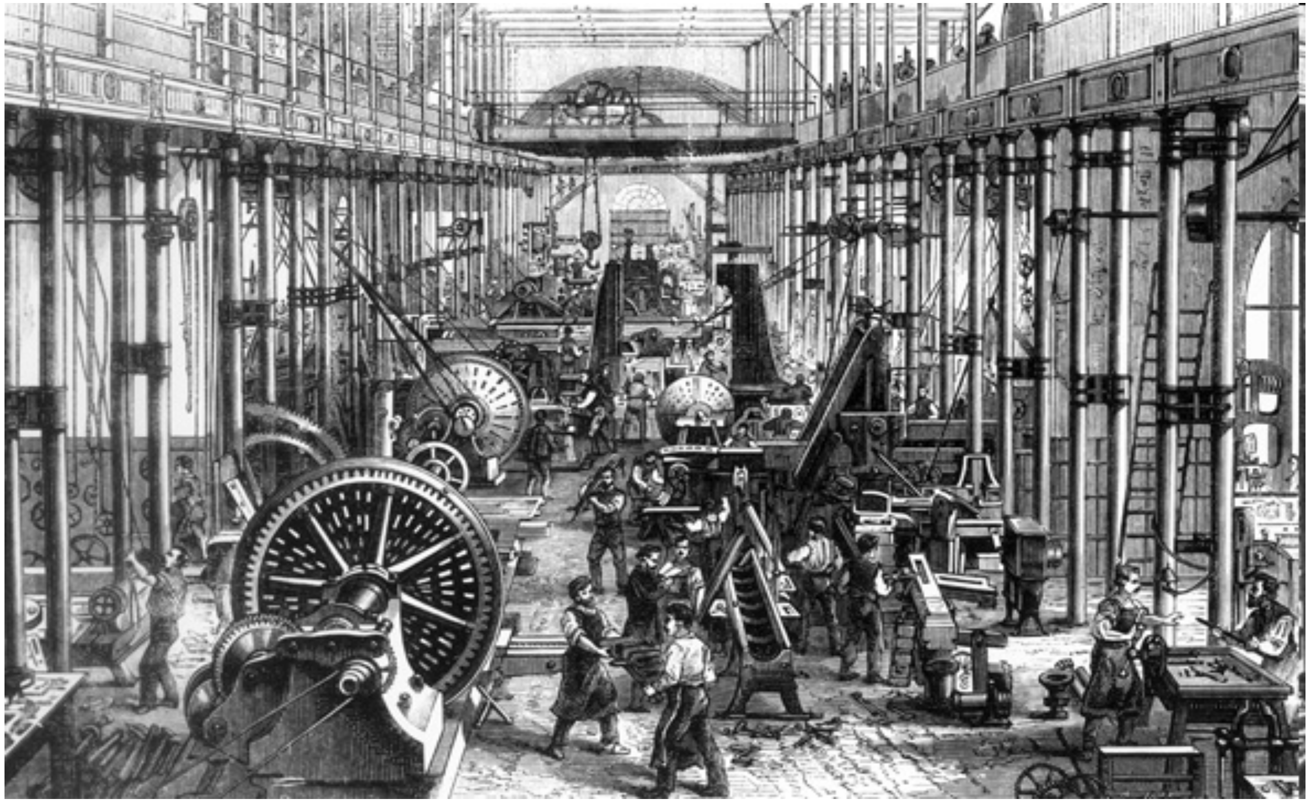
#wso2 #qconlondon

Moore's Law for Data

- The amount of data online went from
  - 5 exabytes in 2002
  - 281 exabytes in 2009
- Doubled every 15 months

- You cannot deal with this data growth with the same applications
  - A reasonable conclusion is that the number of applications will double every 15 months too

Application Development is yet to have its Industrial Revolution



The Industrial Revolution was driven by key technologies:

- **Componentization** – making tools and products from re-usable, standardized components.
- Which relied on **standardized metrics/measures** so that components could fit together
- **Factories** – A large clean space where multiple parts of the production process could share light, power and management to create consistency and governance.
- The IT industry is only just now reaching its industrial revolution. **Open Standards** are the metrics, **Modularity Code and SOA** are the componentization. **Platform-as-a-Service** is the Factory.

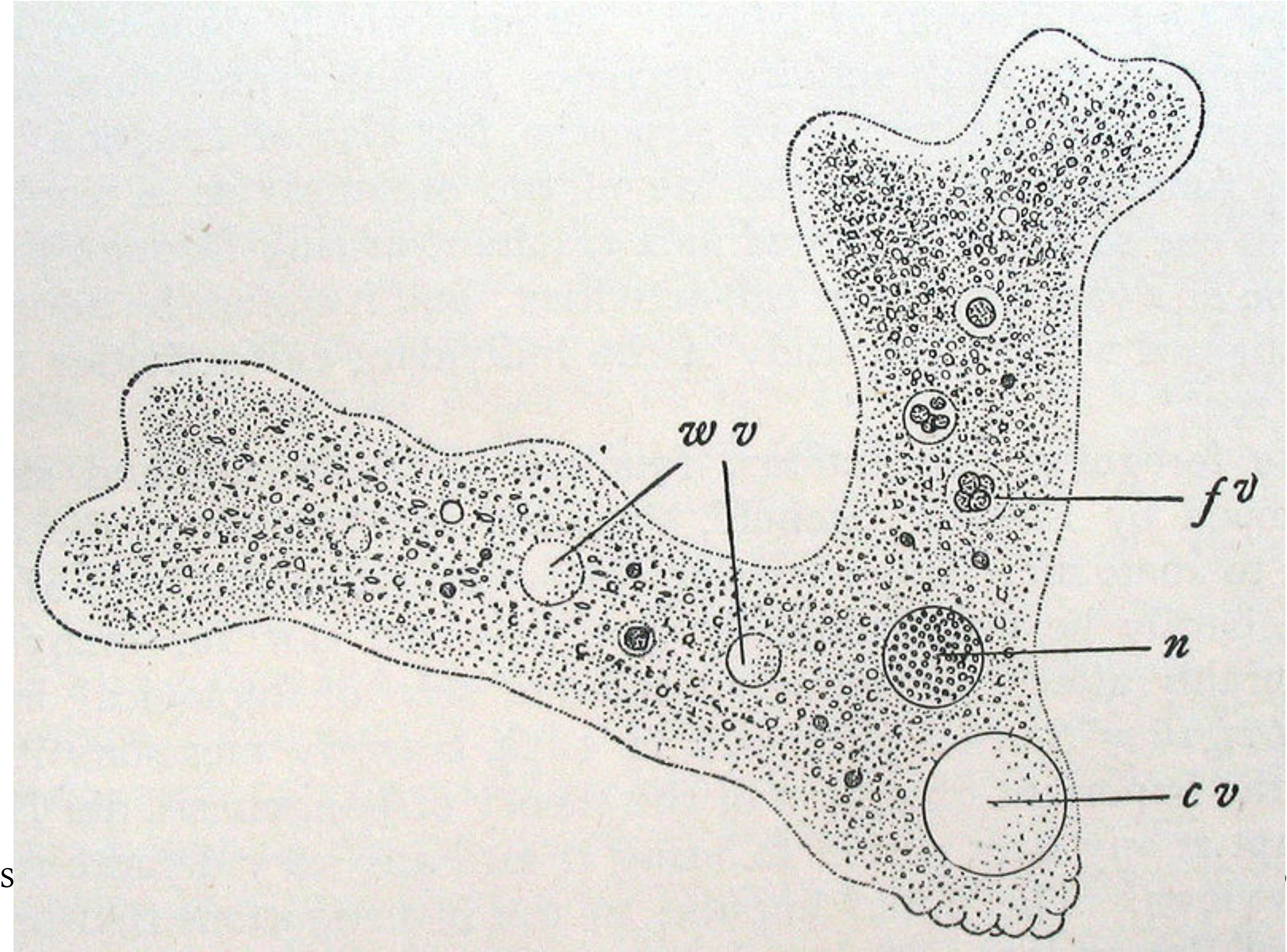
Enterprise Application Development Challenges

- Project Infrastructure takes too long to setup
- Too many projects fall through the cracks and use non-standard infrastructure, build, governance
- Projects use too many different unknown or unapproved libraries, frameworks, etc
- No clear idea of which projects are at which stage
- Few or no metrics on code quality, test coverage, re-use
- Little automated build or test
- Best practices are not applied everywhere
- Infrastructure takes too long to setup

What is Cloud?

- Depends who **you** are
  - **My daughter**: iCloud (her music in the cloud)
  - **My mum**: gmail (her email in the cloud)
  - **My VP sales**: Salesforce (his prospects in the cloud)
  - **Sysadmin**: Amazon/Rackspace/etc (his infrastructure in the cloud)
  - ***:** what *you* care about, self-provisioned, managed, metered and paid per use, in the cloud

Evolution Stage 1
“Cloud Washing”



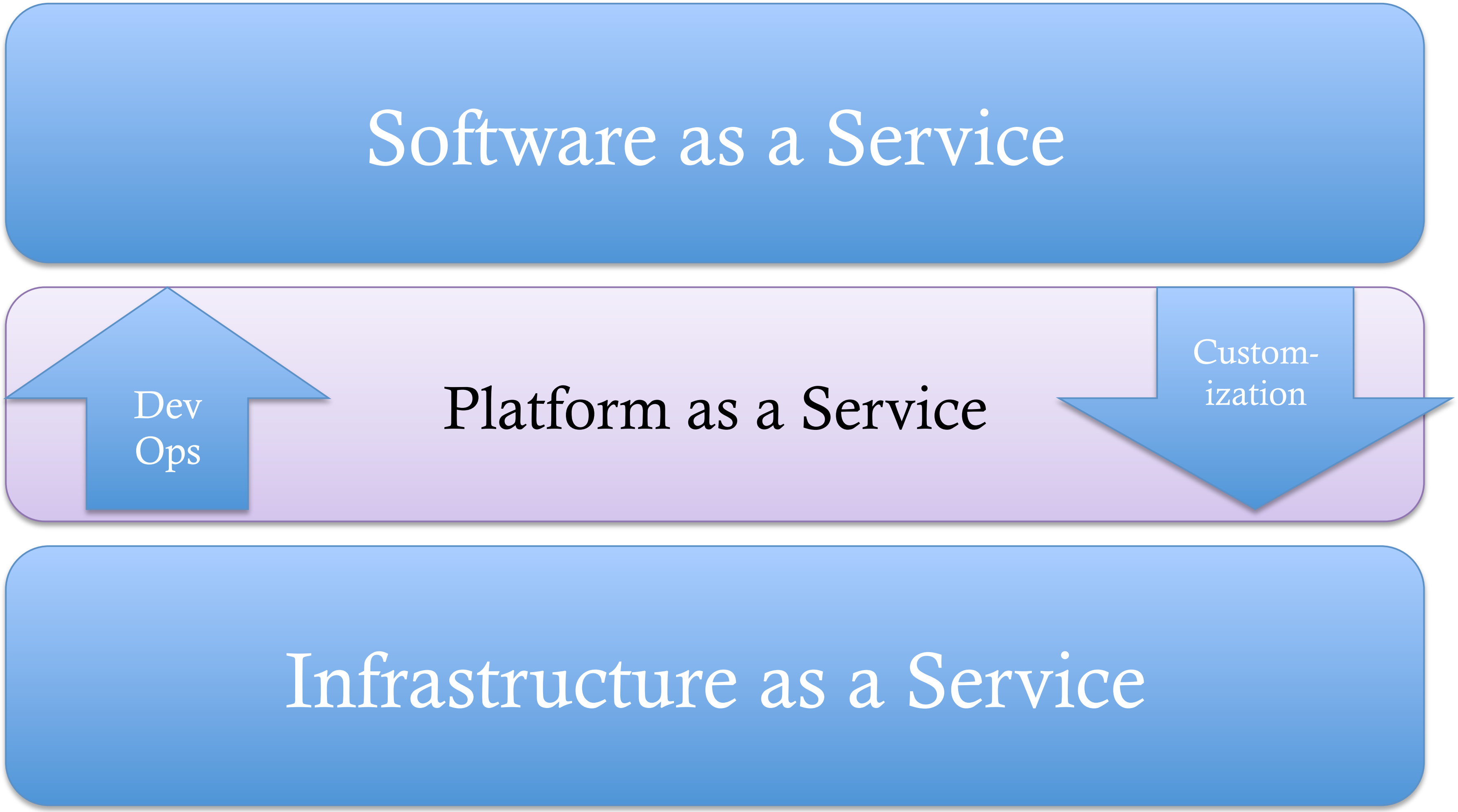
Cloud Native

- **Distributed/Dynamically Wired** **(works properly in the cloud)**
  - Supports deploying in a dynamically sized cluster
  - Finds services across applications even when they move
- **Elastic** **(Uses the cloud efficiently)**
  - Scales up and down as needed
  - Works with the underlying IaaS
- **Multi-tenant** **(Only costs when you use it)**
  - Virtual isolated instances with near zero incremental cost
  - Implies you have a proper identity model
- **Self-service** **(in the hands of users)**
  - De-centralized creation and management of tenants
  - Automated Governance across tenants
- **Granularly Billed and Metered** **(pay for just what you use)**
  - Allocate costs to exactly who uses them
- **Incrementally Deployed and Tested** **(seamless live upgrades)**
  - Supports continuous update, side-by-side operation, in-place testing and incremental production

http://www.flickr.com/photos/ladymaggic/

http://www.flickr.com/photos/jurvetson/

PaaS


Software as a Service

Dev Ops

Platform as a Service

Custom-ization

Infrastructure as a Service

Evolving from different strands

- Evolving “upwards” from IaaS
  - Amazon
    - Queuing, Mail, Notification/Events, Databases, Workflow, etc
    - Elastic BeanStalk
- Evolving “downwards” from SaaS
  - Force.com – a place to host additional per-tenant logic
  - Google App Engine
- Evolving “sideways” from middleware Platforms
  - WSO2, Tibco, vmWare, Oracle, IBM

What do I care about (as a dev)?

- My code – running
  - Not a "VM" but a Virtual App Server
- Not just code:
  - I like Queues and Topics, ESB flows, Workflows, Databases, Logs, Portals, etc
- Not just runtime
  - I like SVN, Git, build, continuous integration, code coverage, automated test
  - As a manager of devs I like governance :-)

Evolution Stage 2
Dealing with "Application / Developer" artefacts

What kind of PaaS is it?

- Application PaaS
  - Google App Engine, Heroku, Stratos AS
- Integration PaaS
  - Mule Ion, Stratos ESB
- Process PaaS
  - ArisOnline, Stratos BPS
- Messaging PaaS
  - StormMQ, Stratos MB
- Development PaaS
  - CloudBees
- "Complete" PaaS

Evolution Stage 3
Expanding Services

Open PaaS

- One of the key concerns of enterprise architects is Open-ness
- Lock-in is a huge concern for PaaS
- Open PaaS *should mean:*
  - Open to run in different places
    - Not just a PaaS but a project or product too
  - Open to run on different IaaS infrastructures
    - Layerable
  - Open to run different types of application
    - Not just Java or Ruby
  - Open to new Services
    - Not just applications
  - Open Source?

Who are the players in the PaaS market?

- Those with only Public PaaS (without a Private PaaS)
  - Force.com
  - Heroku
  - Google App Engine
  - Amazon Elastic Beanstalk
- Those with a Private / Public PaaS
  - Tibco
  - Microsoft (nearly)
  - IBM / Oracle – talked about
- Those with an Open Private / Public PaaS
  - SpringSource CloudFoundry
  - WSO2 Stratos

Stratos Overview

- A full middleware platform available as a service, with self service
  - Fast provisioning
- Based on OSGi
  - Modular, componentized, standard
- Multi-tenant, Elastic, Metered and Billed
  - Effective and powerful
- Available under the Apache License
  - Open Source, Open License, Open Development

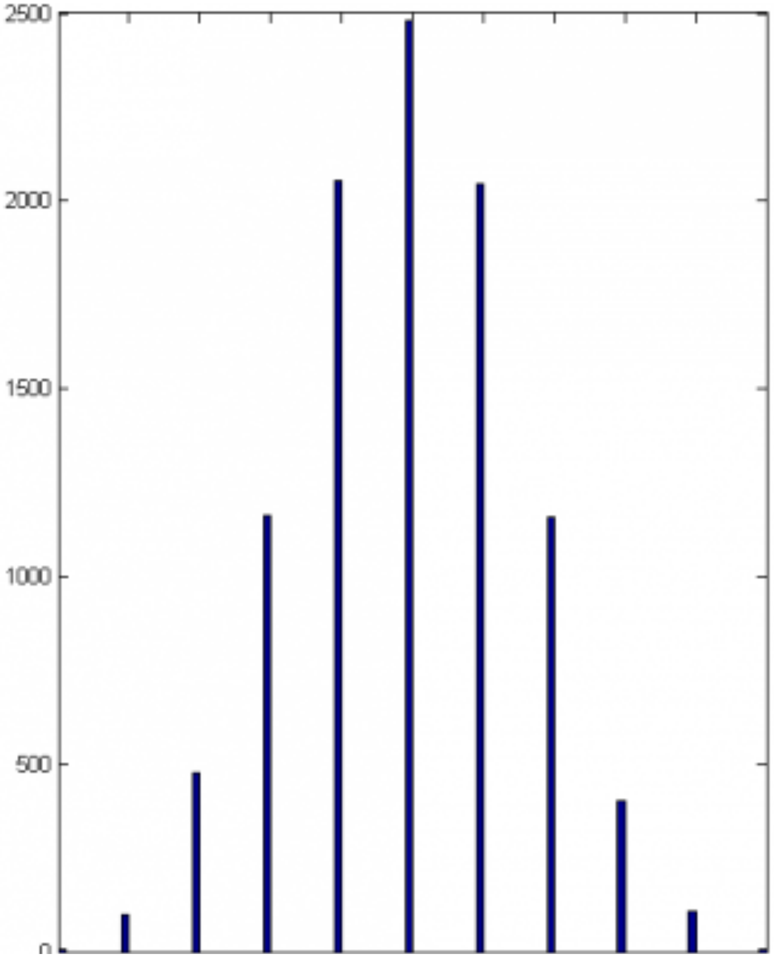
Stratos resources

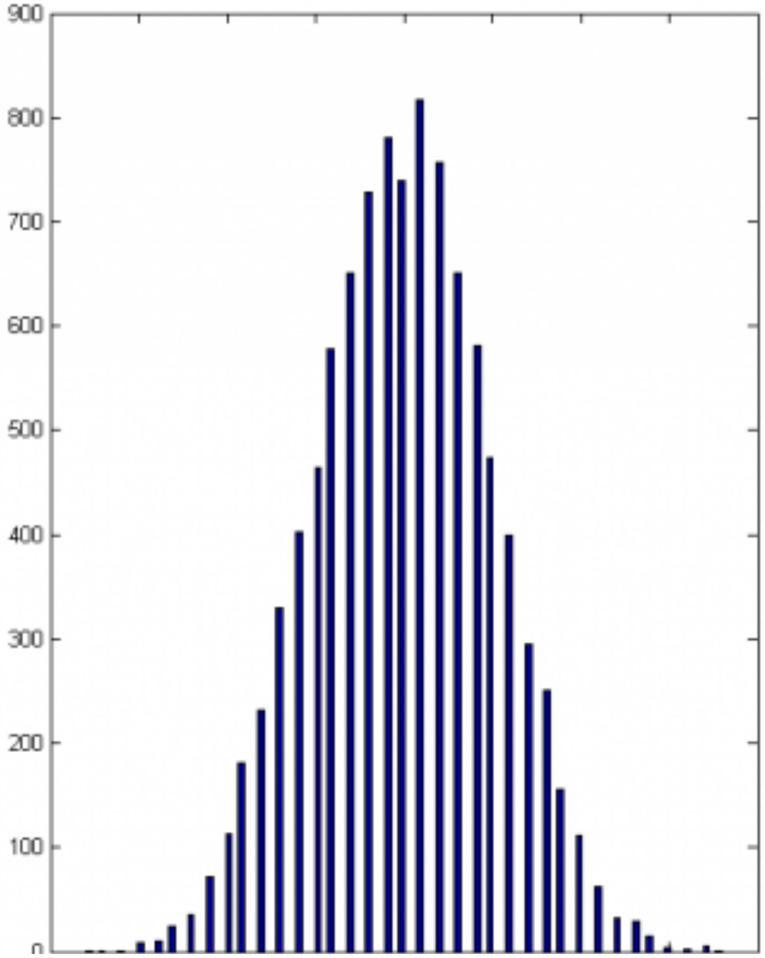
- Stratos SVN
  - http://svn.wso2.org/repos/wso2/trunk/stratos/
- Stratos-dev list
  - https://mail.wso2.org/cgi-bin/mailman/listinfo/stratos-dev
- Stratos 1.5.1 builds
  - http://builder.wso2.org/~carbon/releases/stratos/

StratosLive

http://stratoslive.wso2.com

?


2500
2000
1500
1000
500
0


900
800
700
600
500
400
300
200
100
0

Cloud Economics

- Public Cloud economy is based on the Central Limit Theorem

- For private clouds, the CLT sucks
  - Multi-tenancy is the economy of scale for private clouds

Evolution Stage 4
Dealing with multi-tenancy effectively

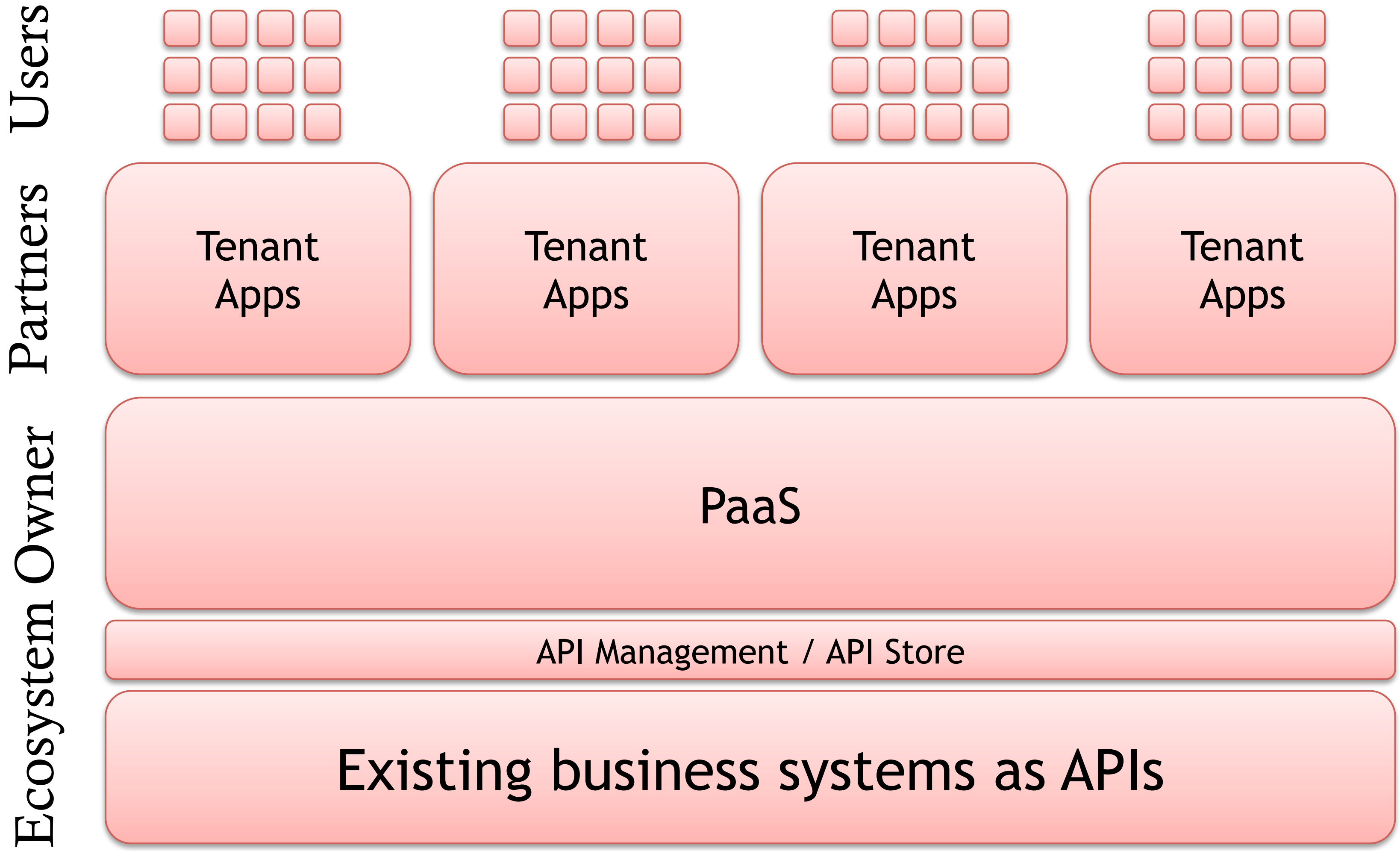
How do you deal with multi-tenancy?

- Separate hardware, separate VMs, separate containers, separate "zones"
- Different techniques for different systems:
  - Java: thread isolation, classloaders, shared classloaders
  - Linux: LXC, chroot
  - MySQL, Oracle: create a DB per tenant
  - Cassandra: keyspace isolation

Run your own PaaS?

- Large enterprise with different groups of developers
  - Consistent runtime internally
  - Better utilization
- Even more interesting:
  - Domain-specific PaaS
  - Vertical PaaS
  - Your Business As A Service

Vertical / Domain PaaS
aka “Your Business As A Service”



Users

Partners

Ecosystem Owner

Tenant Apps

Tenant Apps

Tenant Apps

Tenant Apps

PaaS

API Management / API Store

Existing business systems as APIs

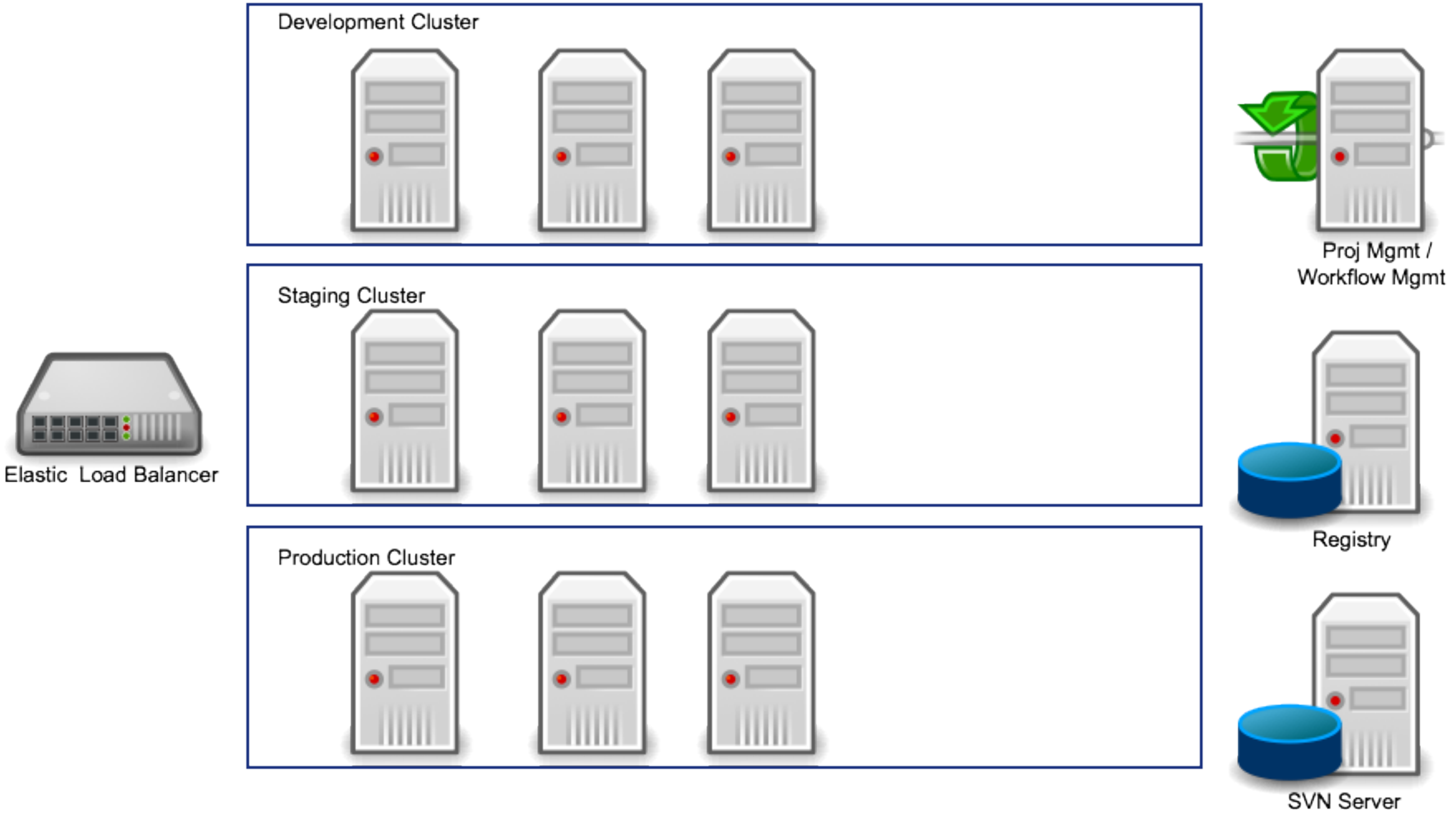
Speciation

- Horizontal
  - – Speciation: Best of Breed PaaS
  - – Despeciation: “Complete” PaaS movement
- Vertical
  - – Speciation: Vertical specific PaaS (Trading, Banking, Betting, Retail, Transcoding, …)
  - – Despeciation: CLT / non-vertical

PaaS will not just be about running code

- Developers care about:
  - Coding
  - Repositories
  - Build
  - Test
  - Approval processes

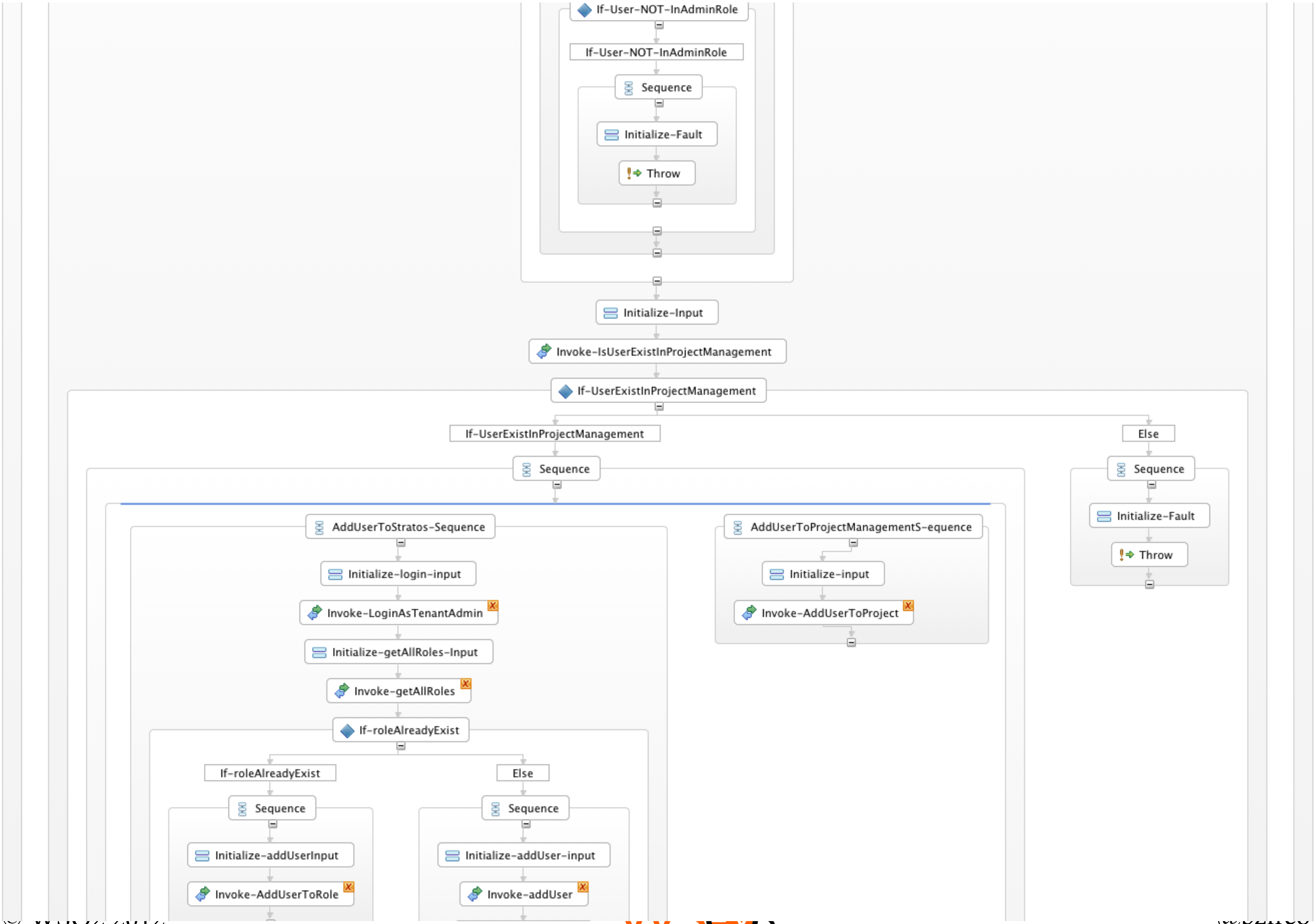
Integrating Development Management into Stratos: App Factory



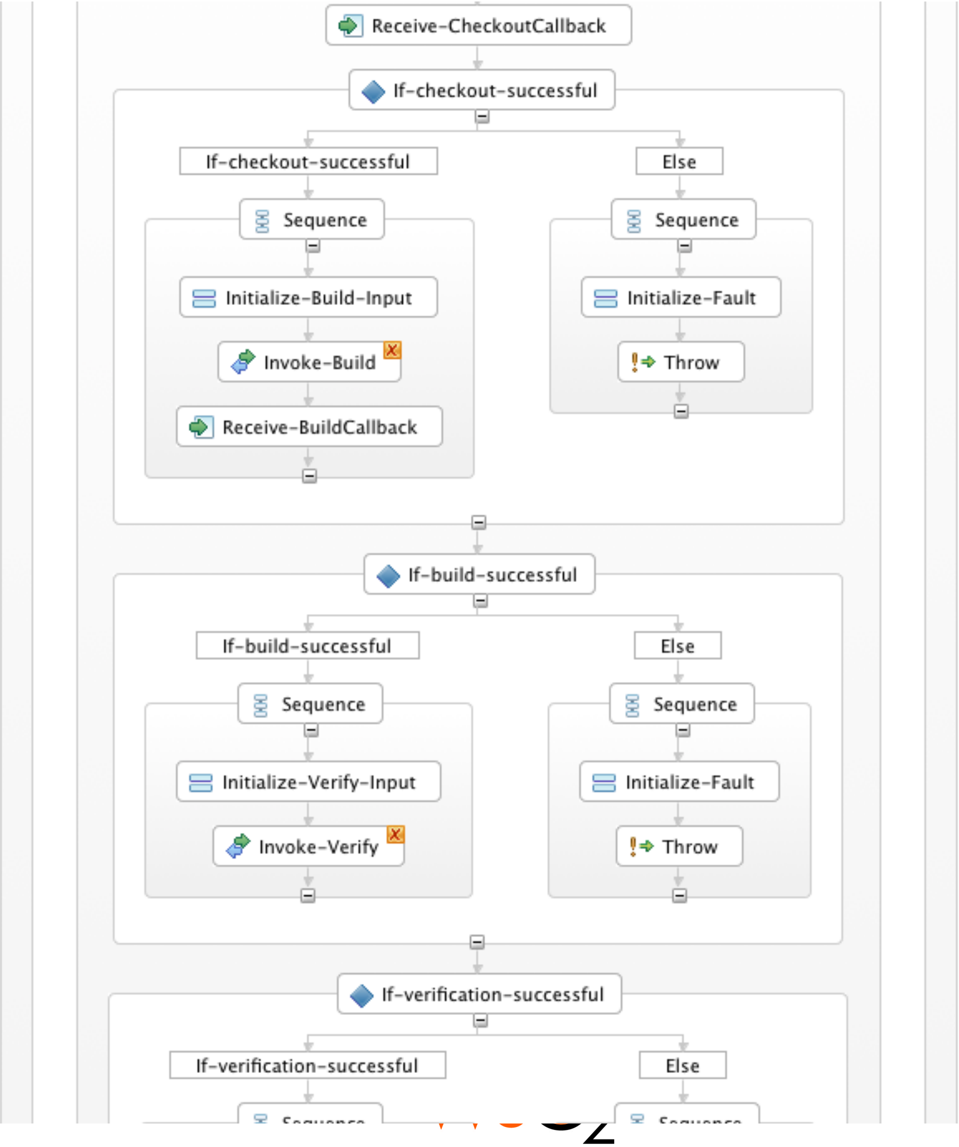
How does it work?

- A set of integrated systems – e.g.:
  - Redmine, project management
  - SVN, Git
  - JIRA
  - Identity Server
  - Governance Registry
  - Maven
  - Stratos
- Integrated using ESB and Business Process Server
  - Each system is connected via ESB
  - Simple BPEL workflows orchestrate activities

AddUserToProject BPEL



DeployAppToDev



Stratos Evolution

- Carbon
  - Evolved from multiple linked projects into a component-ized / kernel-ized system using OSGi
  - Completely Open Source and Open Architecture
- Stratos
  - Initial stage – Elastic Load Balancer
  - Core work:
    - Multi-tenancy in the kernel
    - Self-Service
    - Billing and Metering
  - Next steps
    - Adding more services – messaging, database (SQL/NoSQL), BAM, CEP
- Current work
  - Adding non-Java / Open SPIs for any service
  - Development / Build / Governance

Summary
Evolutionary story

- Self-service deploy and run an application
- From caring about the VM to caring about application-level concerns
- Adding extra runtime services
  - Queues, Databases, Workflows, etc
- Deep multi-tenancy
- Build vertical / domain-specific PaaS
- Full lifecycle from project inception to deployment via build and test

Questions?

Resources

- Try Stratos right now:
  - https://stratoslive.wso2.com/
- Read about Stratos:
  - http://wso2.com/cloud/stratos/
  - Source Download available
- White Paper
  - Selecting Platform as a Service
- Blog Articles
  - What is Platform as a Service?
  - PaaS Evaluation Framework for CIOs and Architects
  - How to simplify Platform as a Service Complexity
  - Searching for Cloud Reference Architecture
- Contact me:
  - paul@wso2.com