



Paul Wallace

9<sup>th</sup> November 2010

Networks on the Ground - The Server Economy

# What is a Yottabyte anyway?

How much?	Exp	Example
Yottabyte	$10^{24}$	Global data in 2020?
Zettabyte	$10^{21}$	Global data in 2010?
Exabyte	$10^{18}$	One day's Internet traffic in 2010
Petabyte	$10^{15}$	Movie post-production
Terabyte	$10^{12}$	PC disk
Gigabyte	$10^9$	PC memory

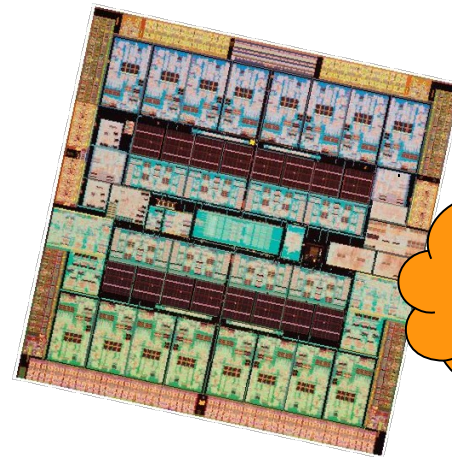
Hands up if you know how much storage you have in your organisation, to the nearest Petabyte? Terabyte?

Where does it all go?

What is stopping us exploiting all this digital wealth?

# To a first approximation...

# ...Computer Hardware is Free



Billions of  
Transistors

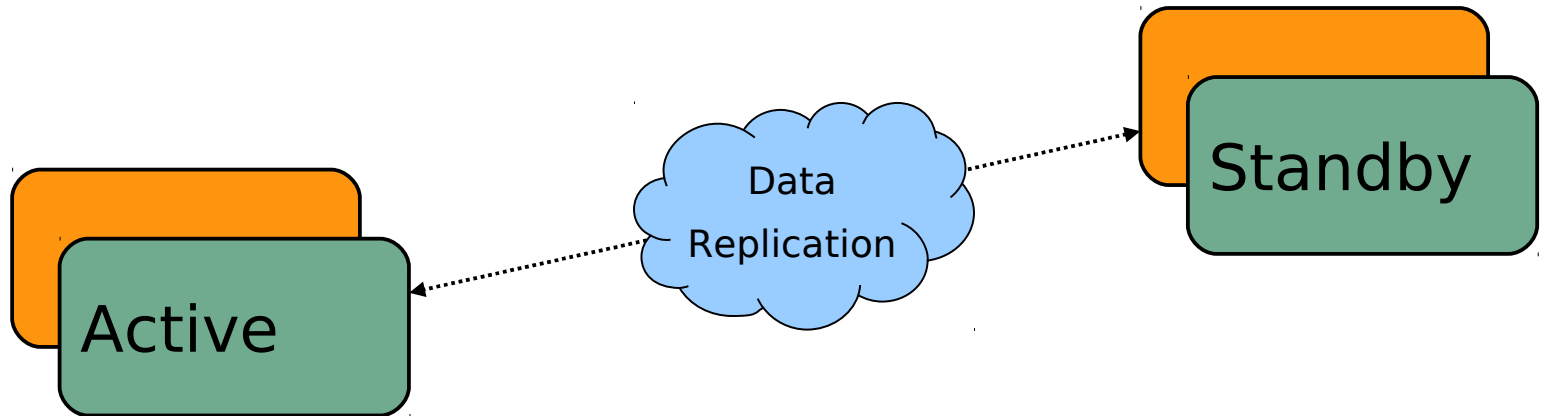
- Multi-core CPUs and dense virtualisation are improving utilisation of assets;
- Multi-GHz clock speeds cut latency;
- Multi-threading leverages memory bandwidth to improve efficiency;

# Look after that data...

Hardware breaks. Buy two of them in a “cluster.”  
In each of two locations. And copy the data again.

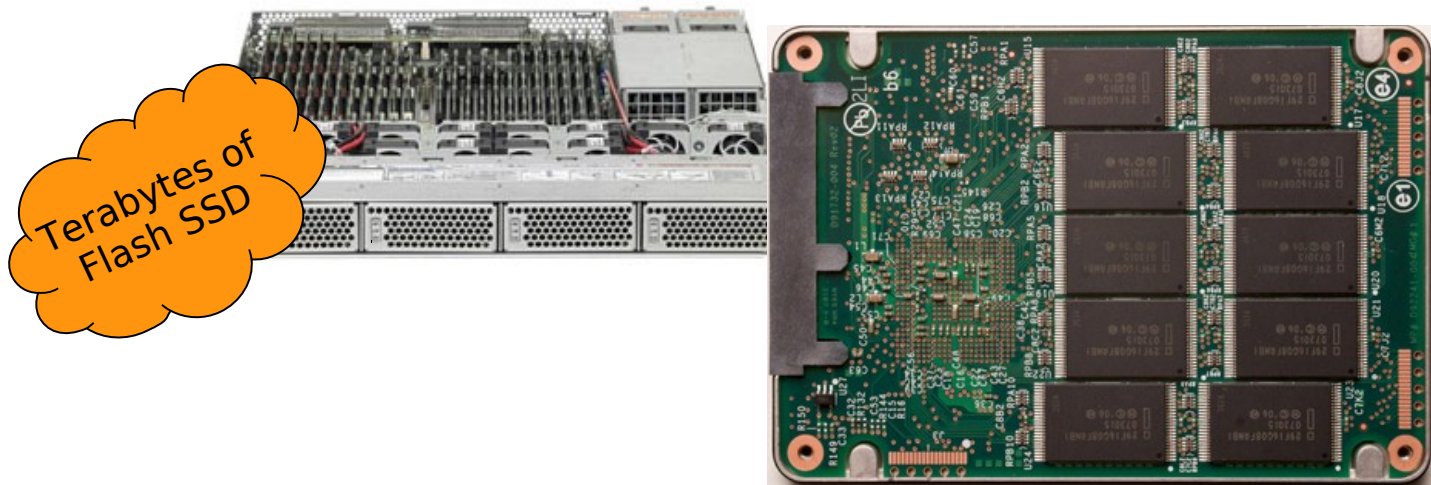
If you're not careful, you end up with **four** times  
as much hardware as you thought you needed.

Don't forget you need 4x power and cooling.



Now... try upgrading your application.  
**Without** taking the service offline.

# The Need for Speed



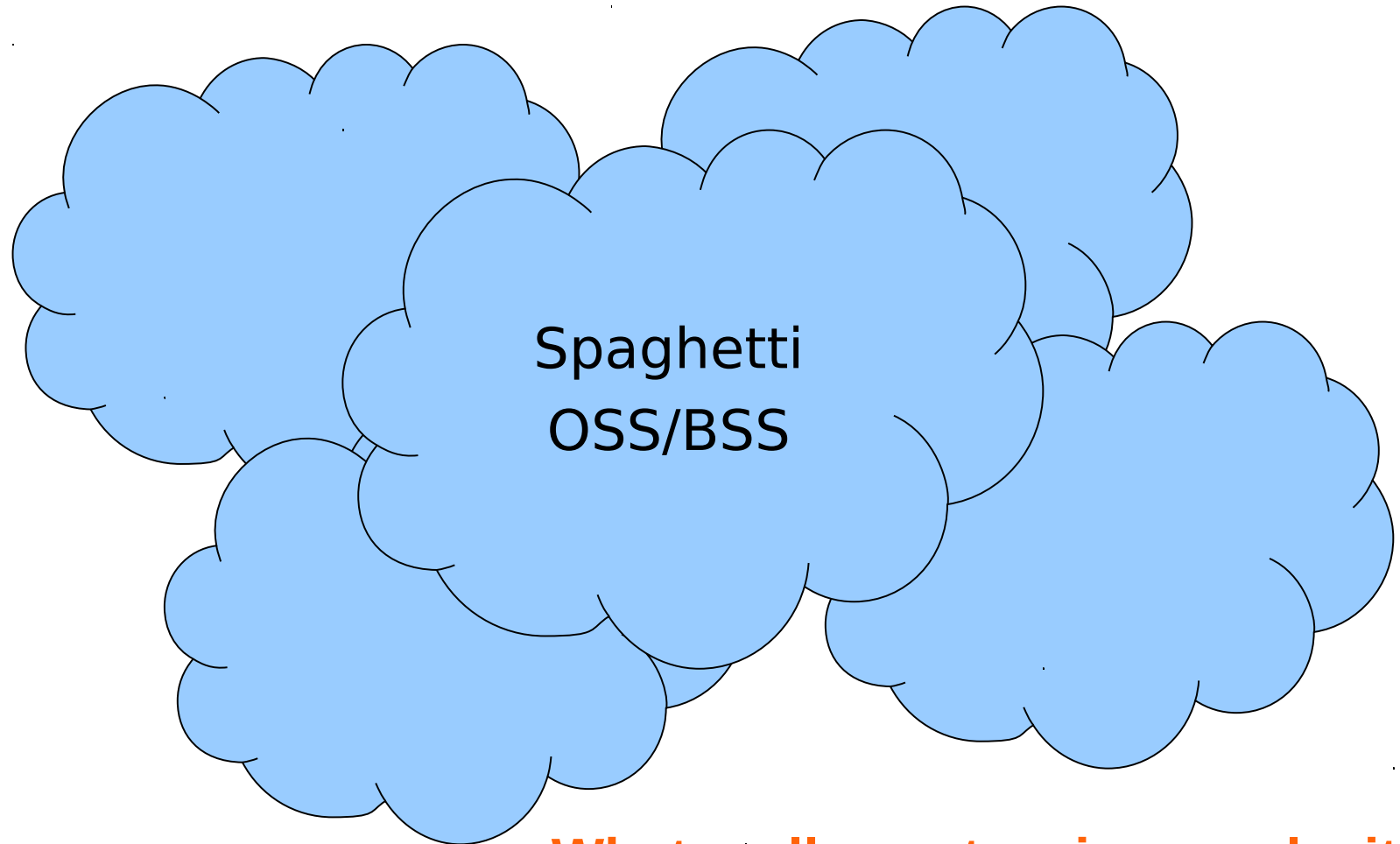
Keep those disks spinning as fast as possible, make sure your data is close at hand.

Or better still, use Solid-state storage, and you can accelerate from 0 to 1,000,000 iops in 1/10<sup>th</sup> of a second.

People are **so** impatient, they want search engines to provide the answer instantly, 24x7x365.

# Does this look familiar?

“Organic” Architecture Blueprint for OSS/BSS:



What really costs... is complexity

# The cost of complexity

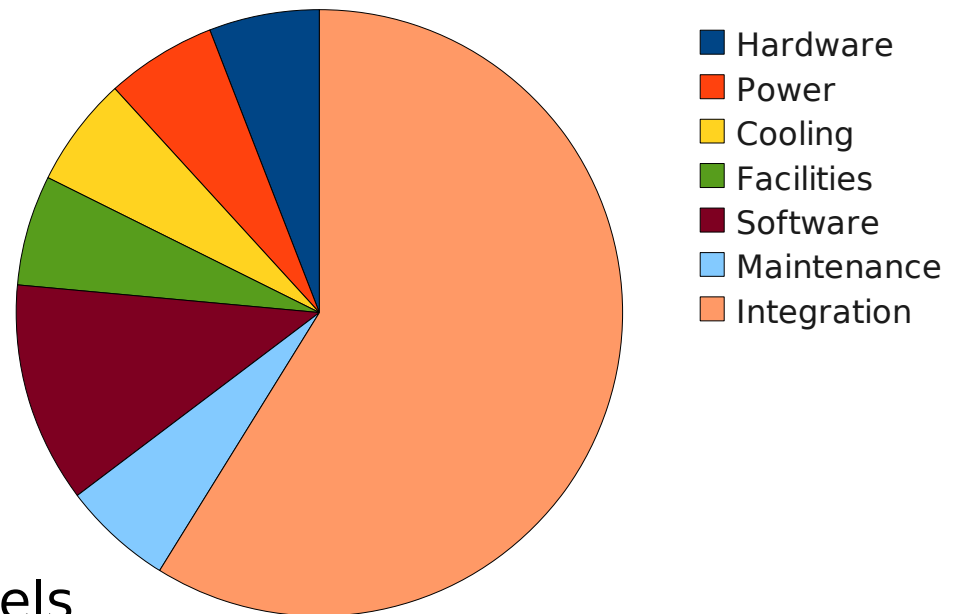
## Typical metric:

“For every \$1 you spend on a new application, you spend another \$10 to make it work with all the other applications.”

Complexity in an organic network is  $O(\#applications^2)$

## Result:

- Brittle systems
- Feature interaction
- Multiple data copies
- Stale data and data models



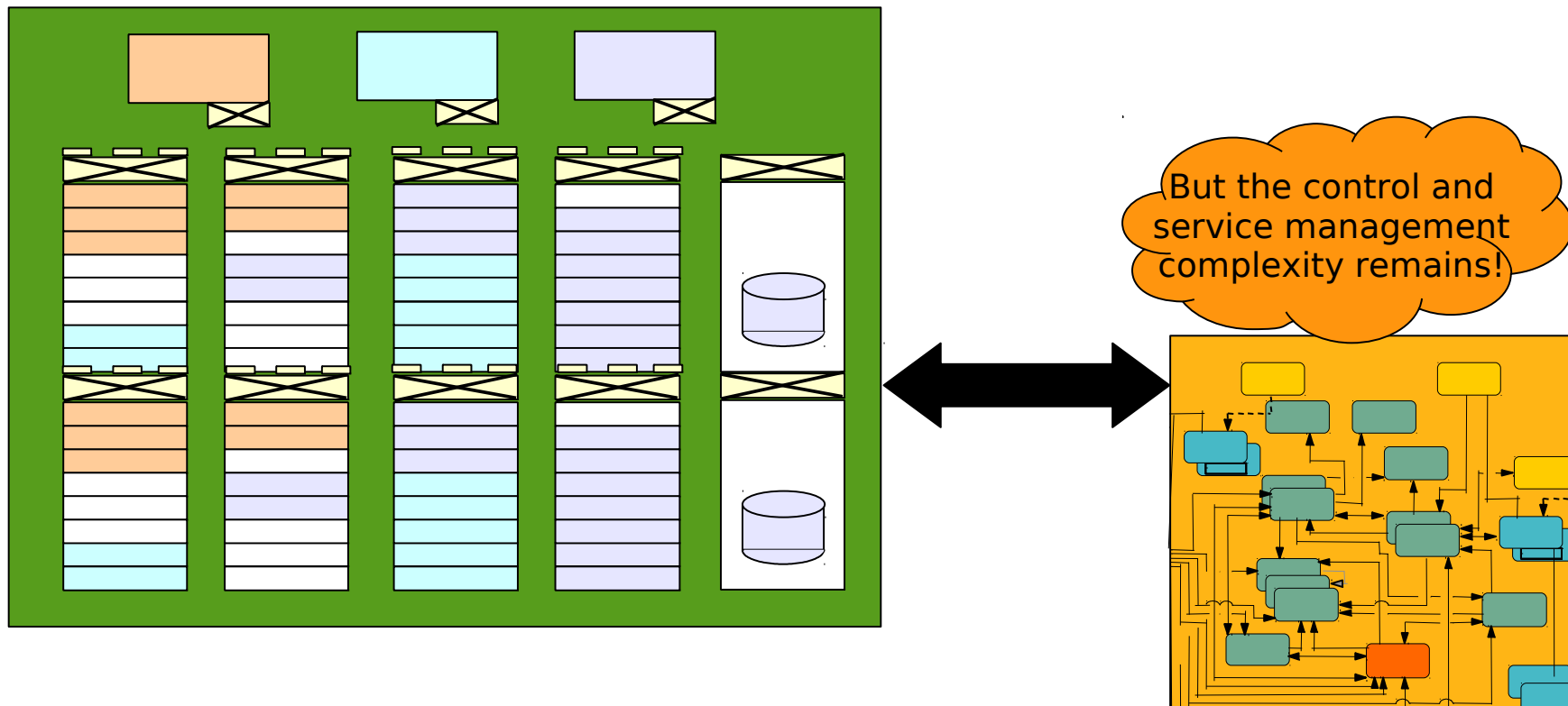
*(Estimated 3-year in-life costs)*

# Clouds. Grids. SaaS. Web Services.

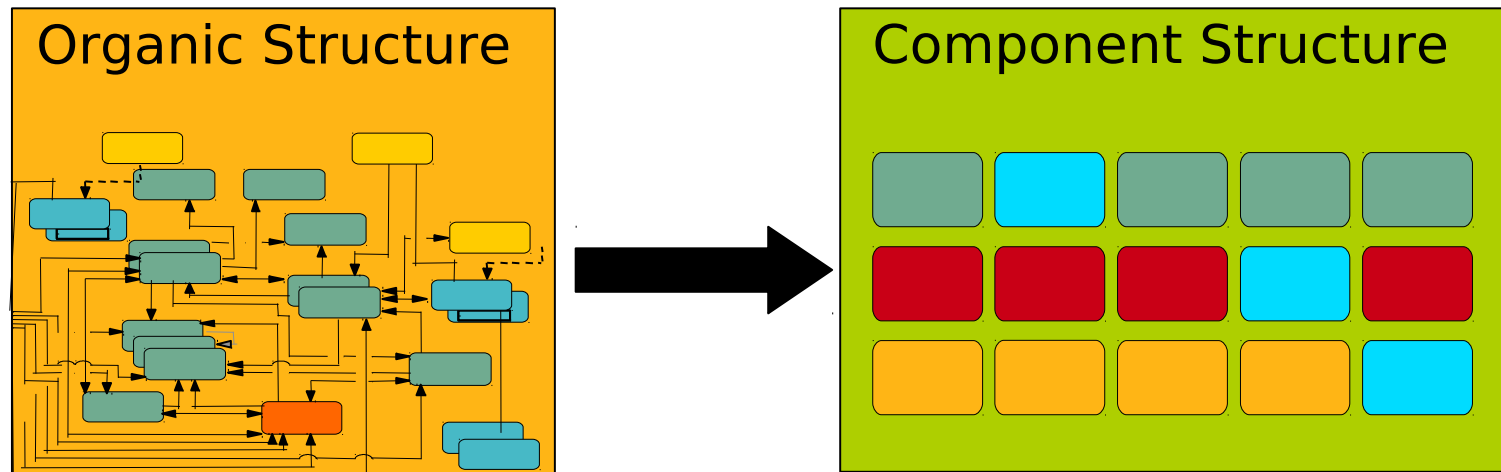
Most service elements **can** be run in a cloud, especially if they scale horizontally. Complexity can be hidden inside the cloud.

Service elements usually map to a simple grid structure.

Complexity in a grid is  $O(\#customers)$



# Taming complexity



Every time you add a new application or service, you may have to change all the others. You can hide it in a cloud, but the logical complexity remains.

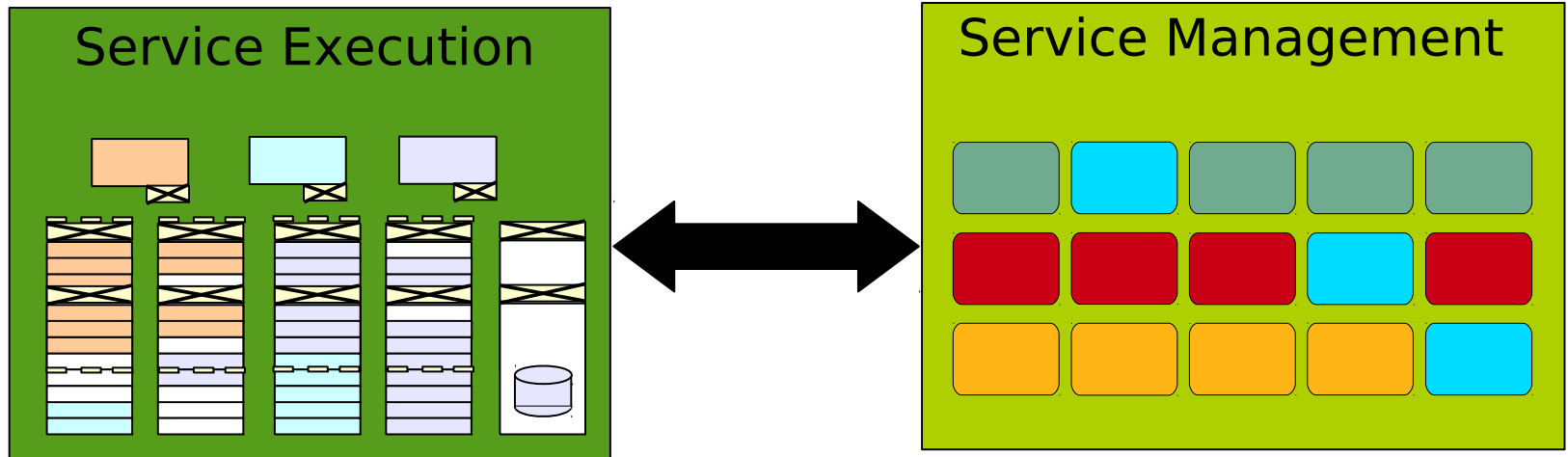
This complexity is actually what differentiates your business!

Component models (eg, TM Forum eTOM) bring an “elasticity” to the system integration, and help reduce the complexity from  $O(N^2)$  to  $O(N)$

# Taming Complexity (2)

Simplified Component Model + Execution Grid shifts the focus from managing complexity to delivering value, and gives much more flexibility for data placement.

Much more control over user content and experience.



**Cloud Computing will not  
transform your business.**

**You must transform your business  
to get the most from a Cloud.**

Paul Wallace  
+44-7802-461169