Progressive Deployment, Experimentation, Multitenancy, No Downtime, Cloud Security, Oh my!

Sam Guckenheimer
Dylan Smith
Microsoft

What We Work on: Azure DevOps

Continuous Delivery for Every Team, Every App, Every Platform

Azure Boards

Deliver value to your users faster using proven agile tools to plan, track, and discuss work across your teams.

Azure Pipelines

Build, test, and deploy with CI/CD that works with any language, platform, and cloud. Connect to GitHub or any other Git provider and deploy continuously.

Azure Repos

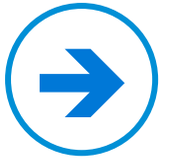
Get unlimited, cloud-hosted private Git repos and collaborate to build better code with pull requests and advanced file management.

Azure Test Plans

Test and ship with confidence using manual and exploratory testing tools.

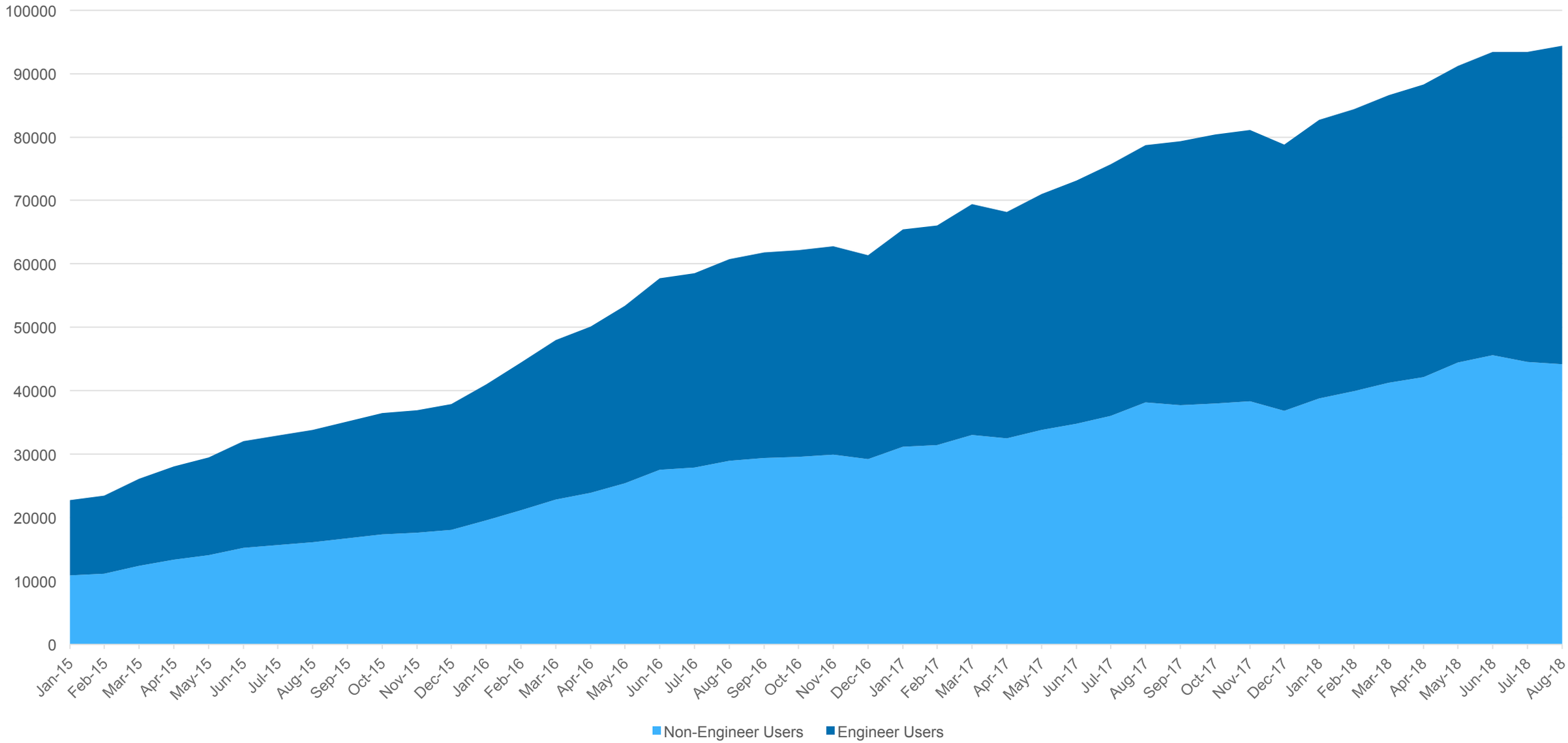
Azure Artifacts

Create, host, and share packages with your team, and add artifacts to your CI/CD pipelines with a single click.



https://azure.com/devops

Azure DevOps: Millions of users + Most of Microsoft



Data: Internal Microsoft engineering system activity, September 2018

DevOps at Scale

Azure DevOps is the toolchain of choice for Microsoft engineering with over 90,000 internal users

https://aka.ms/DevOpsAtMicrosoft

372k
Pull Requests per month

4.4m
Builds per month

5m
Work items viewed per day

78,000
Deployments per day

2m
Git commits per month

500m
Test executions per day

500k
Work items updated per day

Data: Internal Microsoft engineering system activity, September 2018

2018 State of DevOps Report Says…

COMPARING THE ELITE GROUP AGAINST THE LOW PERFORMERS, WE FIND THAT **ELITE PERFORMERS HAVE...**

46 TIMES MORE
frequent code deployments

2,555 TIMES FASTER
lead time from commit to deploy

7 TIMES LOWER
change failure rate
(changes are 1/7 as likely to fail)

2,604 TIMES FASTER
time to recover from incidents

Presented by

DORA
DEVOPS RESEARCH & ASSESSMENT


https://screeningfilm.com/events/shh-at-the-movies/
What do you do about your existing busin code?

Our Journey to DevOps

Sprint 1
August 2010

VSTS Preview
Sprint 29
June 2012

VSTS GA
Sprint 64
April 2014

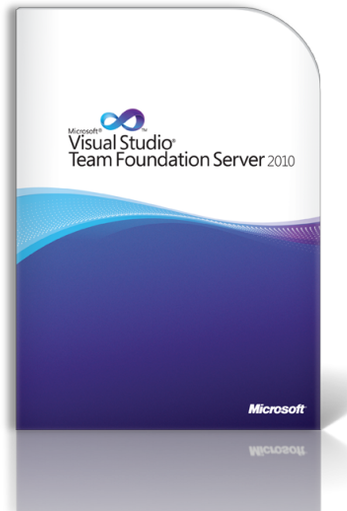
1ES
Sprint 67
June 2014

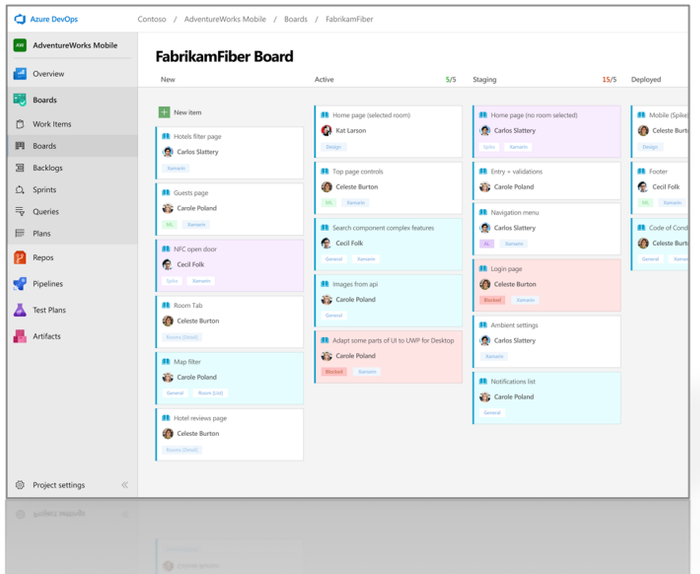
Extensions
Sprint 92
Nov 2015

GVFS
Sprint 102
June 2016

Azure DevOps
Sprint 142
Oct 2018

2010 | 2011 | 2012 | 2013 | 2014 | 2015 | 2016 | 2017 | 2018





Where We Started

Constraint: Maintain compatibility between cloud and on-prem

How do you start? Re-architect first then Move to Cloud OR Move to Cloud first then Re-architect [improve]

2010 – we were just using Scrum

We had TFS

Load balancer compatible

“Team Project Collections” isolated security

Server framework

No experience with Azure

All data in SQL

Only AD identities

No organizations/accounts

We had no experience as a cloud service.

TFS assumed

Single tenant

Offline upgrades

No telemetry

No deployment cadence

No live site culture, on-call

A journey of a thousand miles begins with a single sprint

Microsoft Azure

Moved On-Prem to One instance, but …

Not scalable

Every new account created a database

11k+ account databases broke SQL

Not affordable

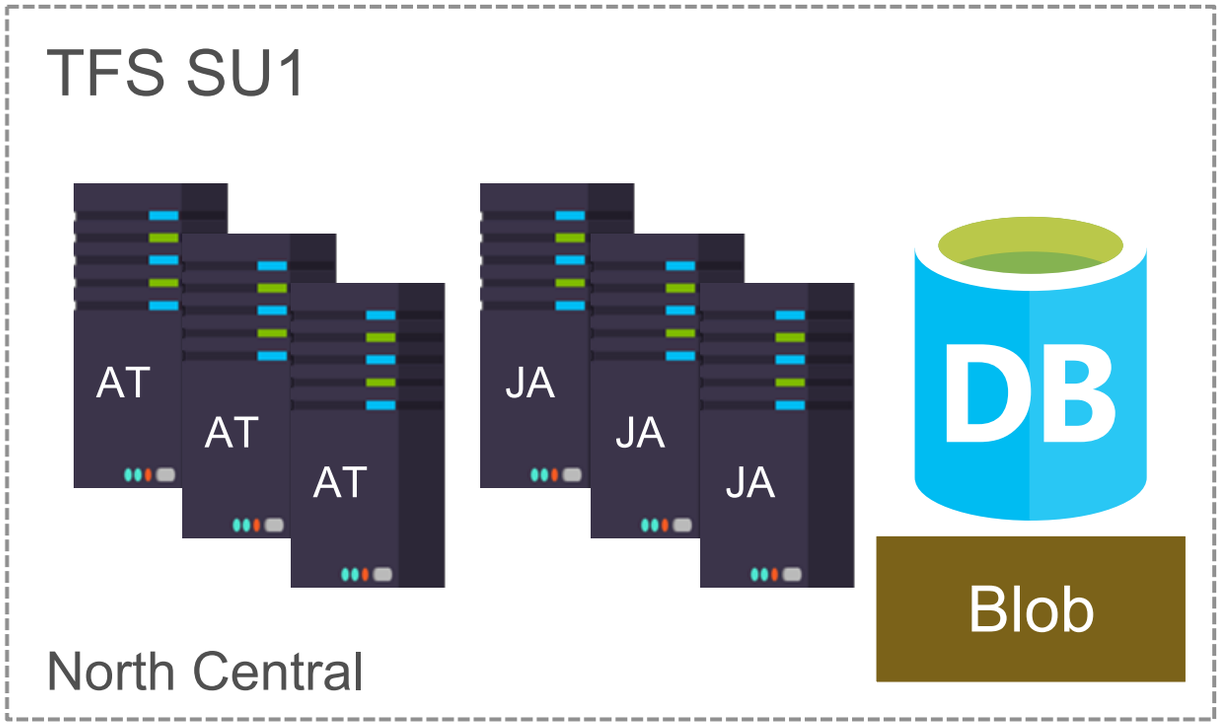
Most of the service is free

COGS unacceptable

Downtime not acceptable

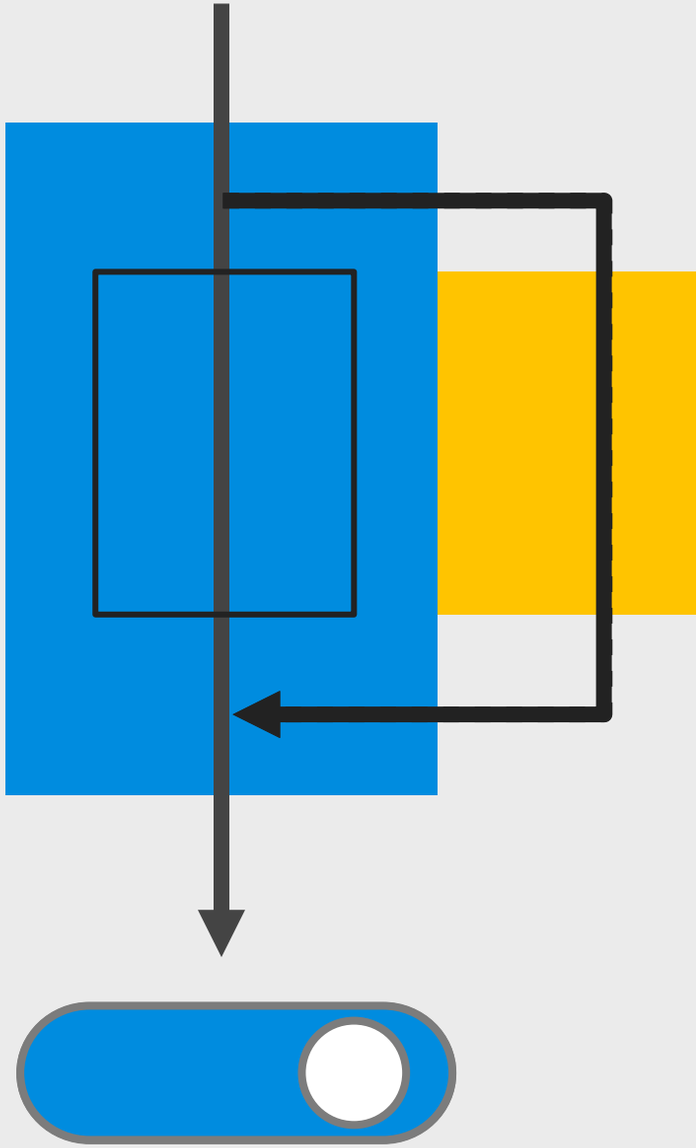
Every update required maintenance window

Service is global…no dark time



Feature Flags

- All code is deployed, but feature flags control exposure
  - Reduces integration debt
- Flags provide runtime control down to individual user
- Users can be added or removed with no redeployment
- Mechanism for progressive experimentation & refinement
- Enables dark launch



Demo

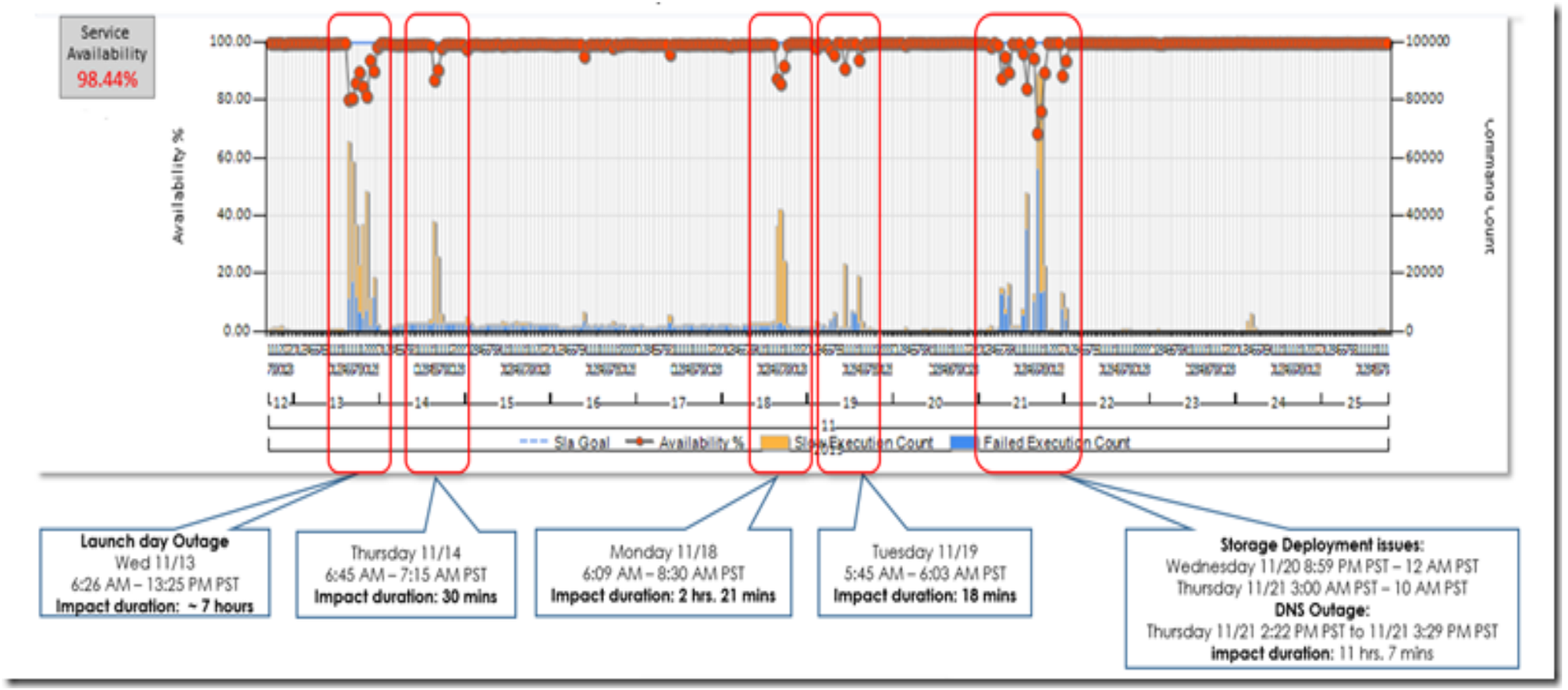
Feature Flags in Production

Microsoft Azure

That's great…what could go wrong?

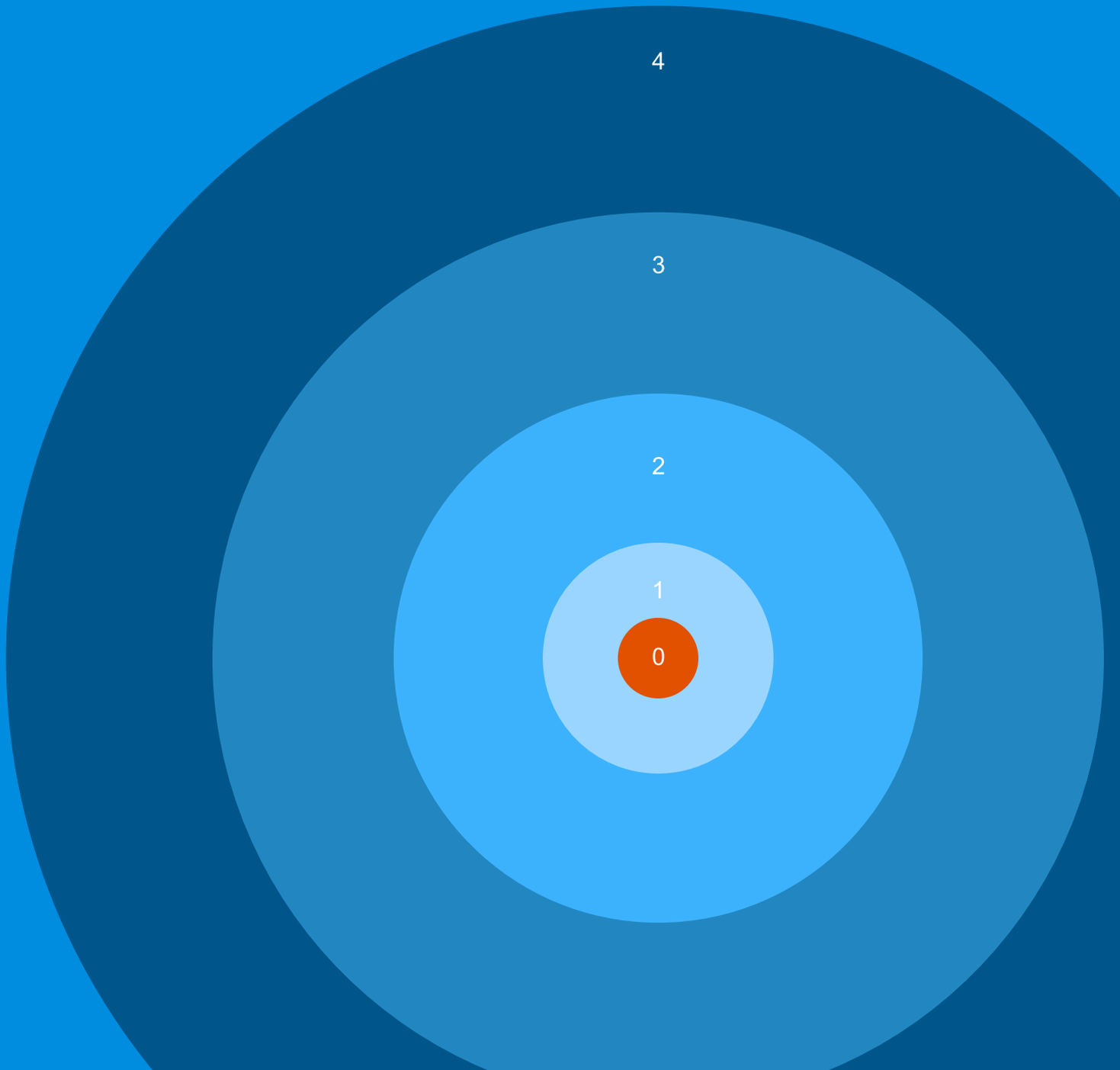
Features to be revealed at Connect 2013 event

We turned features on globally (SU1) just before the keynote...



Your aim won’t be perfect.

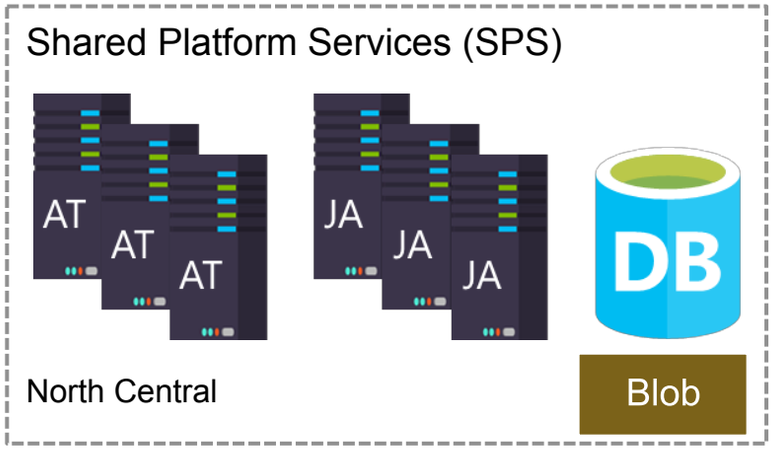
Control the blast radius.

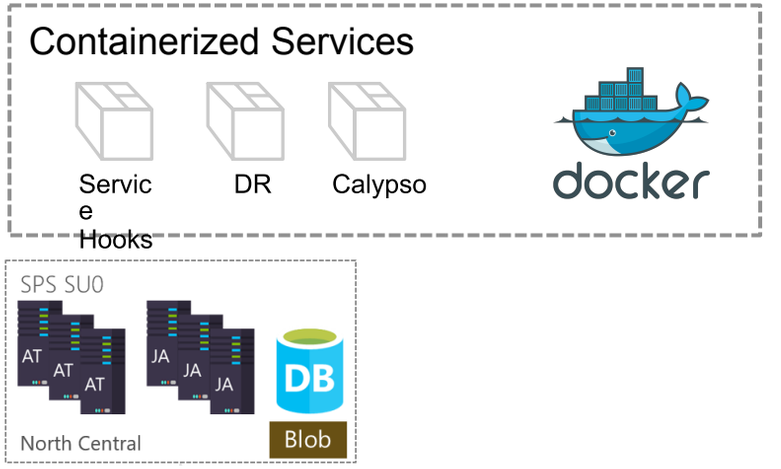


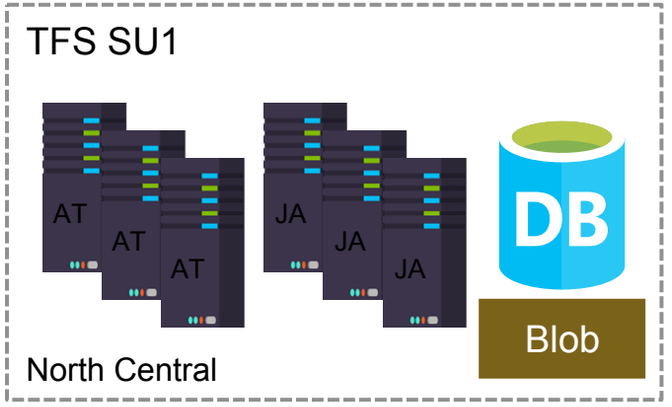
Multiple Data Centers with incremental roll out

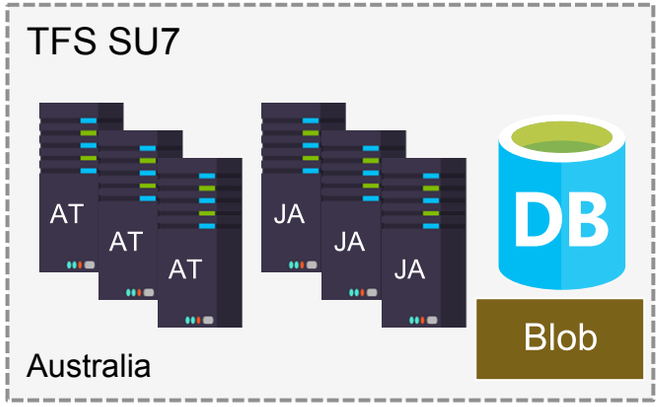
TFS
- Git/Version Control
- Work Item Tracking
- Build & Release
- Test

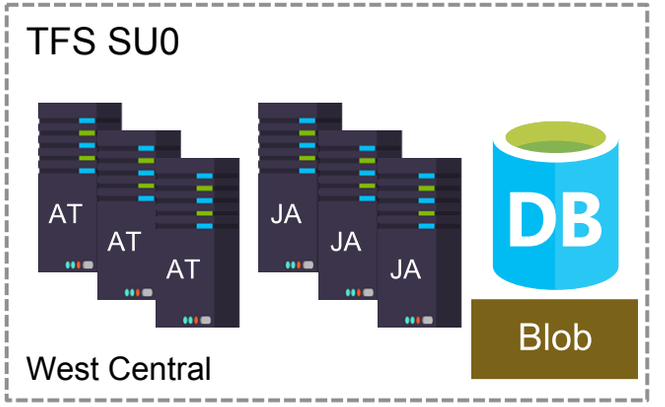
SPS: common services
- Account
- Identity
- Profile
- Licensing











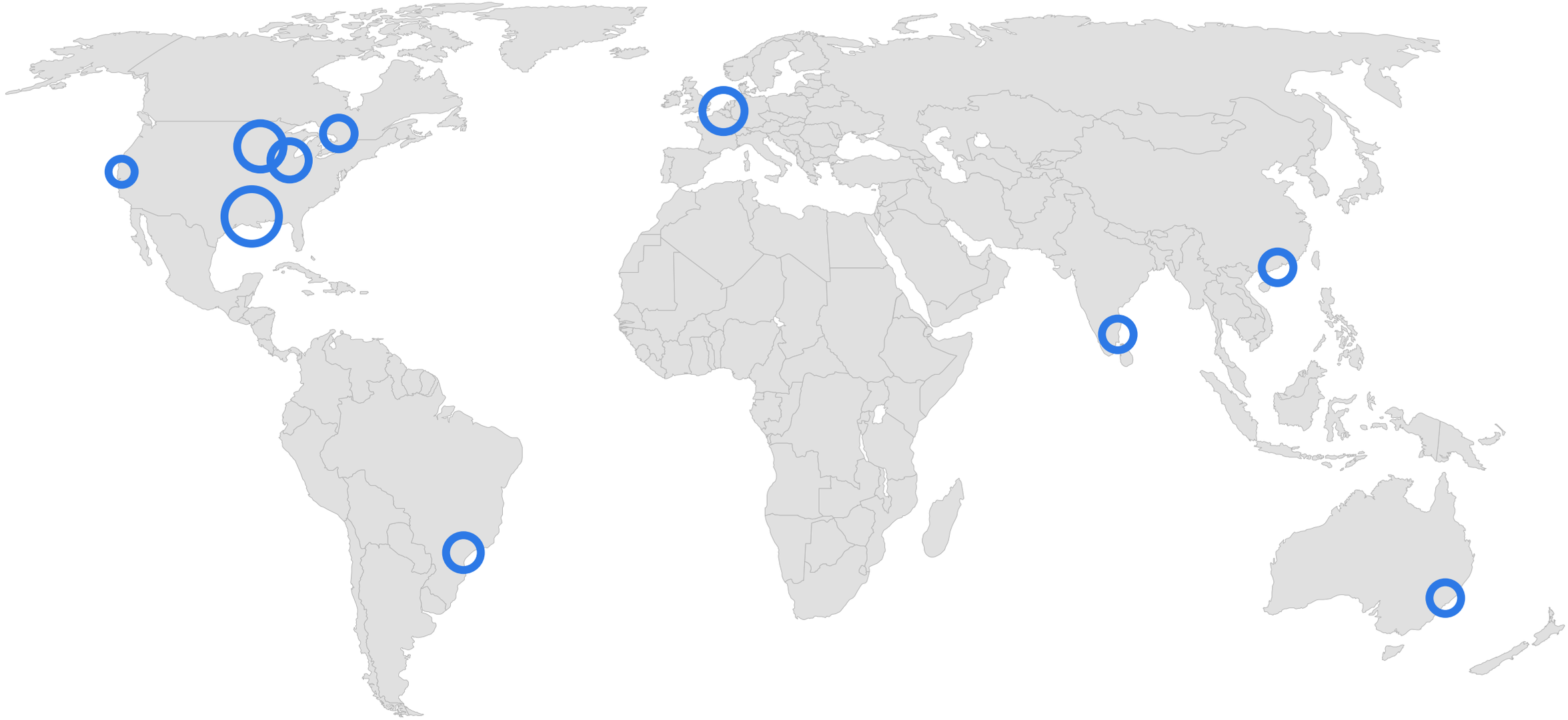
Demo

Realworld Progressive Deployment

Microsoft Azure

Get Out of a Single Data Center

Customers today are in 229 countries and all 7 continents



Managing the pipeline:

How do you

go fast

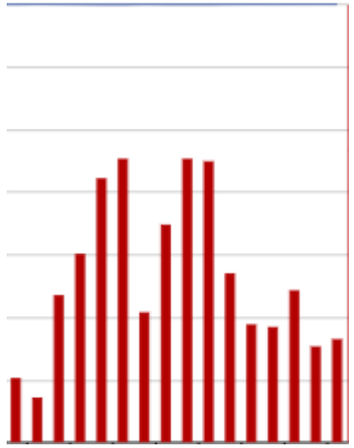
and

not

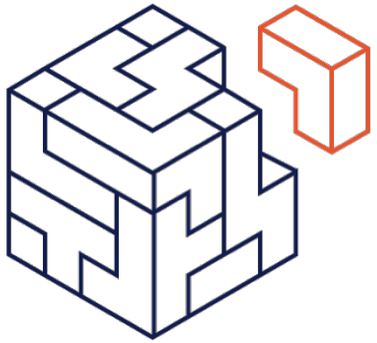
break things?

Resiliency Patterns

Testing in Production



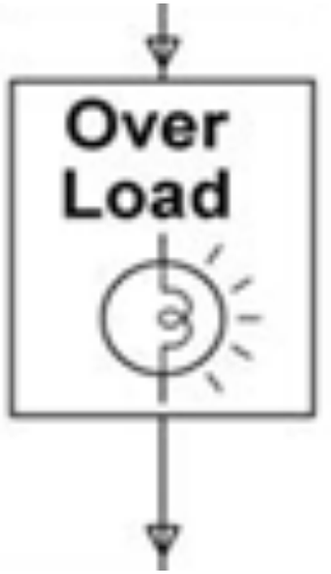
Microservice Isolation



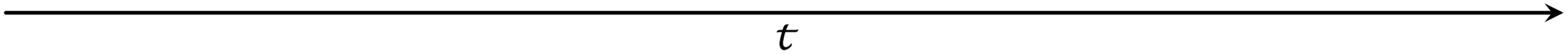
Observability

```
request
Context
.Trace(
103085,
.
```

Circuit Breakers



Throttling



Demo

Microservices Versions in Practice

Microsoft Azure

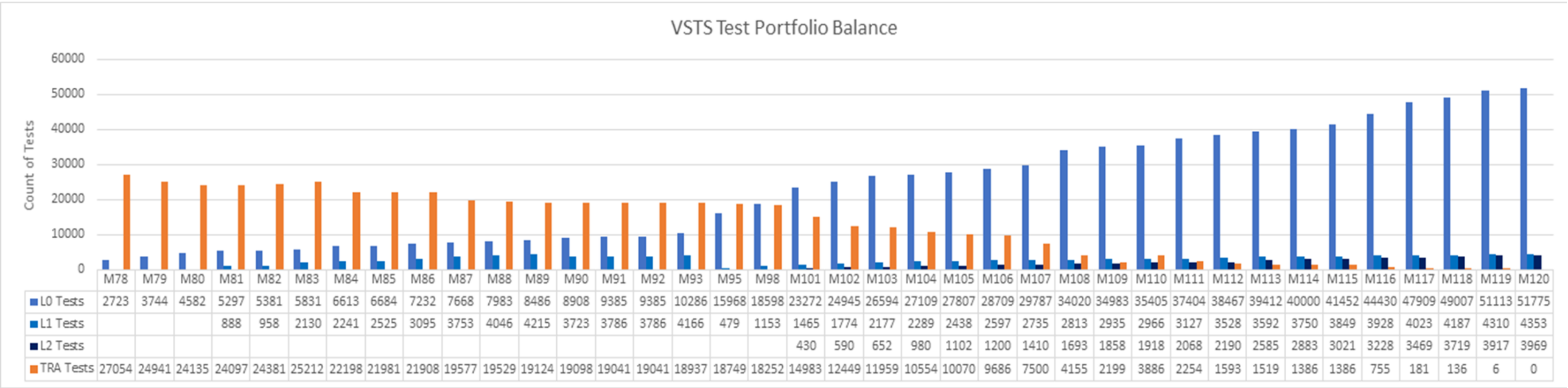
Sprint over sprint progress creating new and retiring old tests

L0 – Requires only built binaries, no dependencies

L1 – Adds ability to use SQL and file system

Run L0 & L1 in the pull request builds

L2 – Test a service via REST APIs



	M78	M79	M80	M81	M82	M83	M84	M85	M86	M87	M88	M89	M90	M91	M92	M93	M95	M98	M101	M102	M103	M104	M105	M106	M107	M108	M109	M110	M111	M112	M113	M114	M115	M116	M117	M118	M119	M120
L0 Tests	2723	3744	4582	5297	5381	5831	6613	6684	7232	7668	7983	8486	8908	9385	9385	10286	15968	18598	23272	24945	26594	27109	27807	28709	29787	34020	34983	35405	37404	38467	39412	40000	41452	44430	47909	49007	51113	51775
L1 Tests				888	958	2130	2241	2525	3095	3753	4046	4215	3723	3786	3786	4166	479	1153	1465	1774	2177	2289	2438	2597	2735	2813	2935	2966	3127	3528	3592	3750	3849	3928	4023	4187	4310	4353
L2 Tests																			430	590	652	980	1102	1200	1410	1693	1858	1918	2068	2190	2585	2883	3021	3228	3469	3719	3917	3969
TRA Tests	27054	24941	24135	24097	24381	25212	22198	21981	21908	19577	19529	19124	19098	19041	19041	18937	18749	18252	14983	12449	11959	10554	10070	9686	7500	4155	2199	3886	2254	1593	1519	1386	1386	755	181	136	6	0

Demo

Using Pull Requests to test changes before commit to Master

Microsoft Azure

There’s no place like production

Azure DevOps

Definition of Done

Live in production, collecting telemetry, supporting or diminishing the starting hypothesis.

Microsoft

Security Mindset - Assume Breach

Started with war games to the learn attacks and practice response

VS.

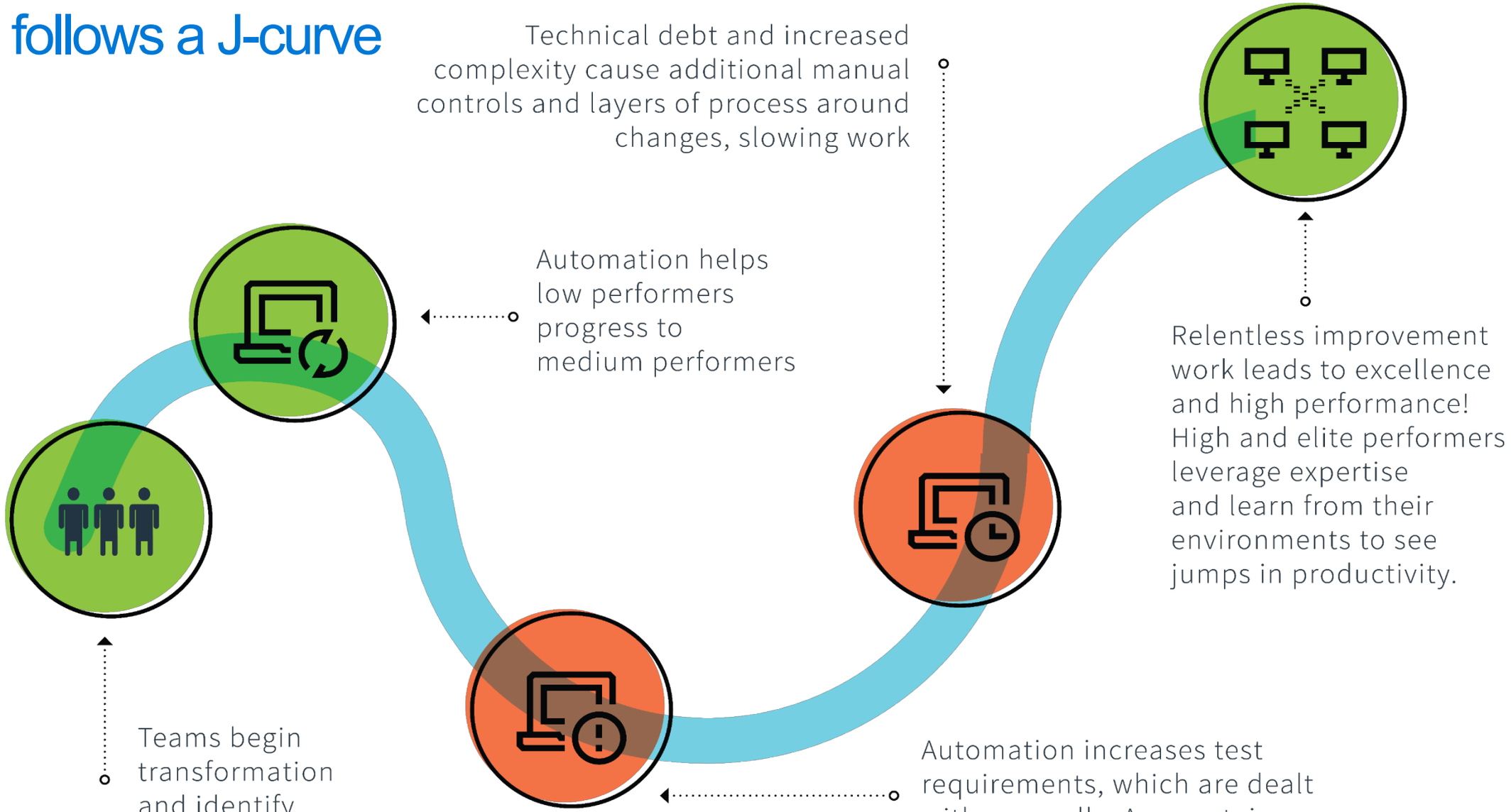
- Initially double-blind test
- Over time, eliminated blue team

Our defenders need to be our defenders

Shifted left to prevent top risks

- Credential theft
- Secret leakage
- OSS vulnerabilities

Progress follows a J-curve



Lessons we learned

You *can* refactor incrementally and you must

Safe deployment requires controlled exposure

Continuous improvement requires continuous experiments

Trunk-based development requires fast, reliable testing

Green should mean green, red should mean red

You’re not done until telemetry confirms you’re done

Change takes time – but you won’t get there unless you start

Thank you!

Sam Guckenheimer
Dylan Smith

@SamGuckenheimer
@Dylan_Smith

https://aka.ms/DevOps/