Microsoft

Sharing our journey
Microsoft's DevOps Transformation

Anthony Borton
DevOps Architect
Microsoft DevOps CAT

Anthony Borton

DevOps Architect – DevOps Customer Advisory Team

Microsoft MVP in Dev Tools - 11 years

Microsoft Certified Professional - since ’93

Microsoft Certified Trainer - since ’96

Professional Scrum Master

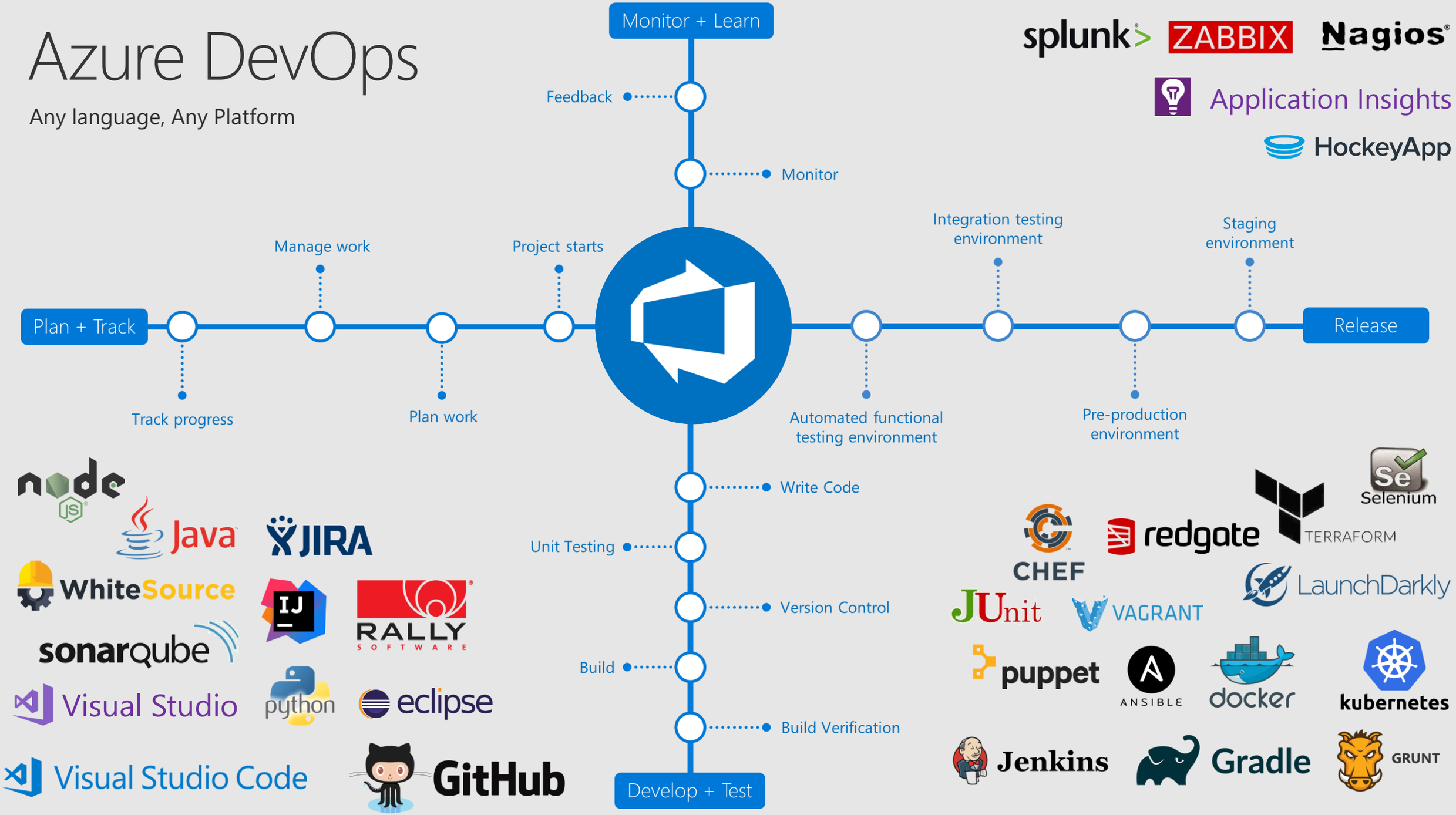
Professional Scrum Developer

Australia’s first Professional Scrum Developer Trainer

@AnthonyBorton

Azure DevOps

Any language, Any Platform



splunk> ZABBIX Nagios

Application Insights

HockeyApp

node JS

Java

JIRA

WhiteSource

IJ

RALLY SOFTWARE

sonarqube

Visual Studio

python

eclipse

Visual Studio Code

GitHub

CHEF

redgate

TERRAFORM

Selenium

JUnit

VAGRANT

LaunchDarkly

puppet

ANSIBLE

docker

kubernetes

Jenkins

Gradle

GRUNT

DevOps at Microsoft

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

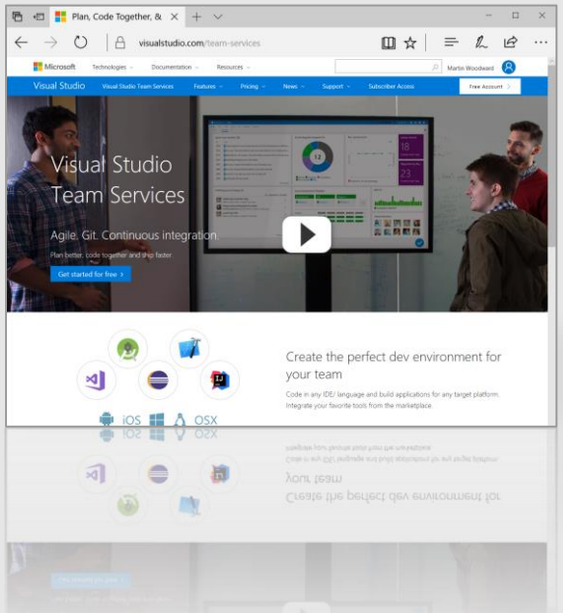
500m
Test executions per day

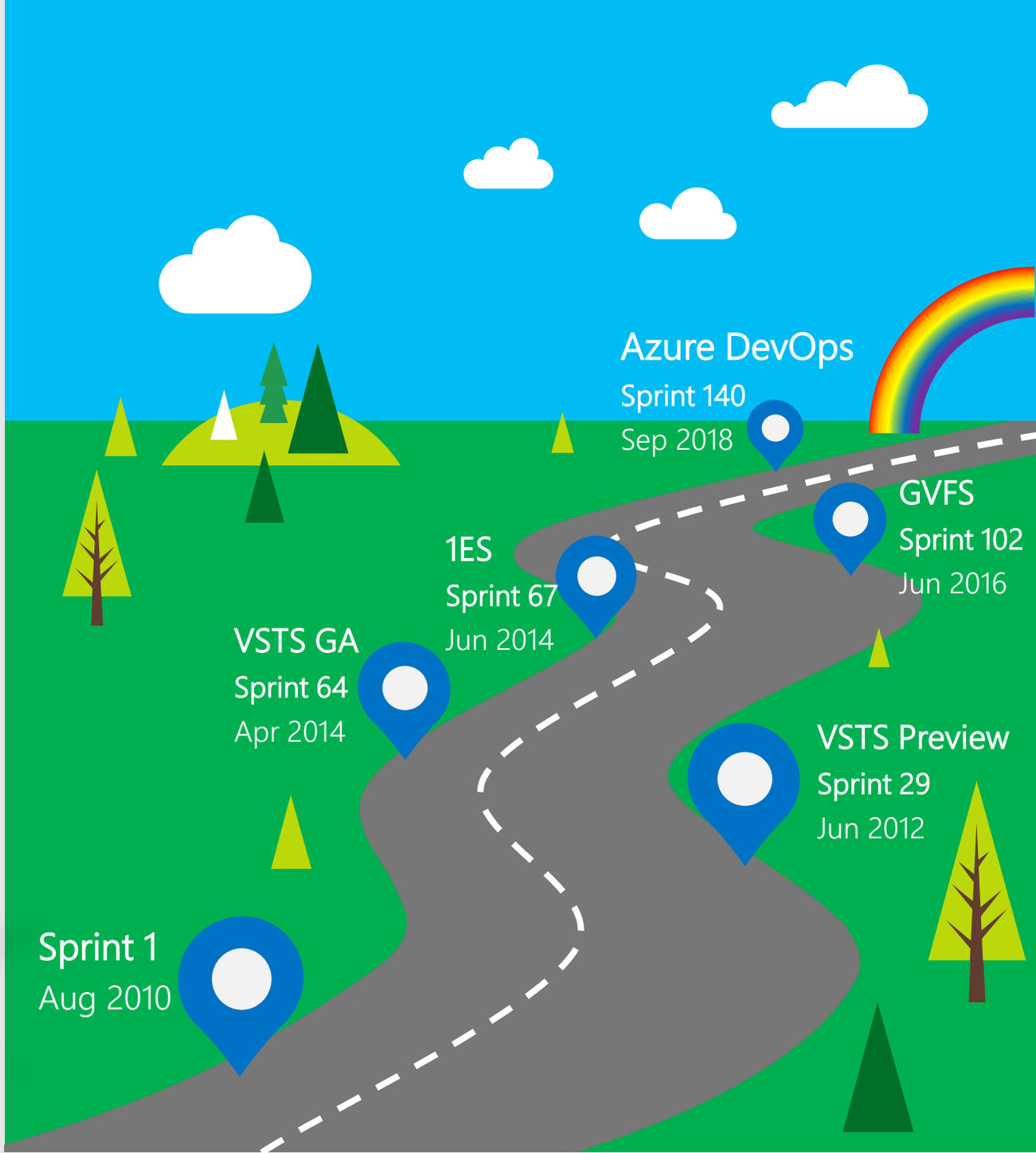
500k
Work items updated per day

@AnthonyBorton

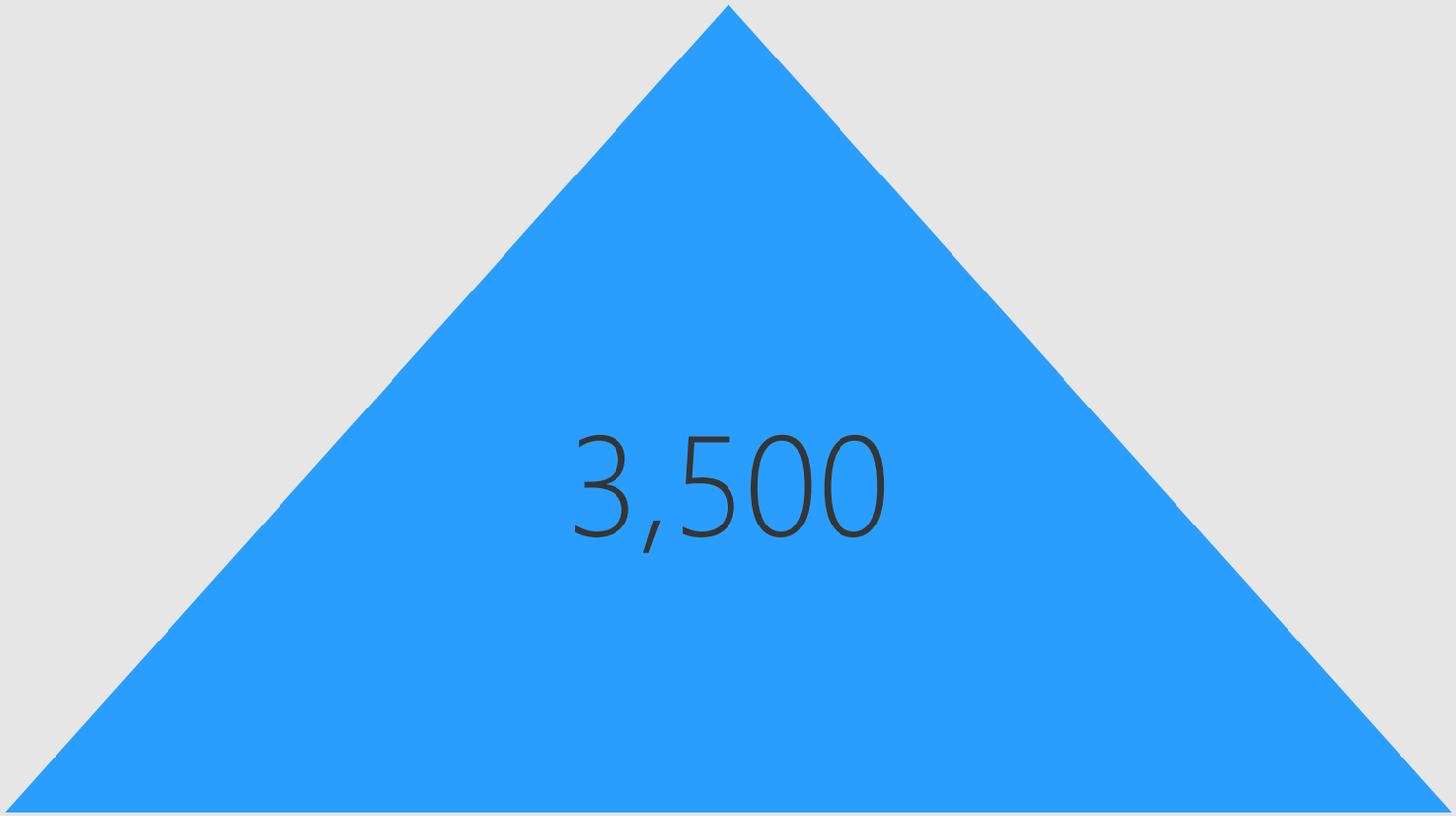
Data: Internal Microsoft engineering system activity, August 2018

The Journey to DevOps

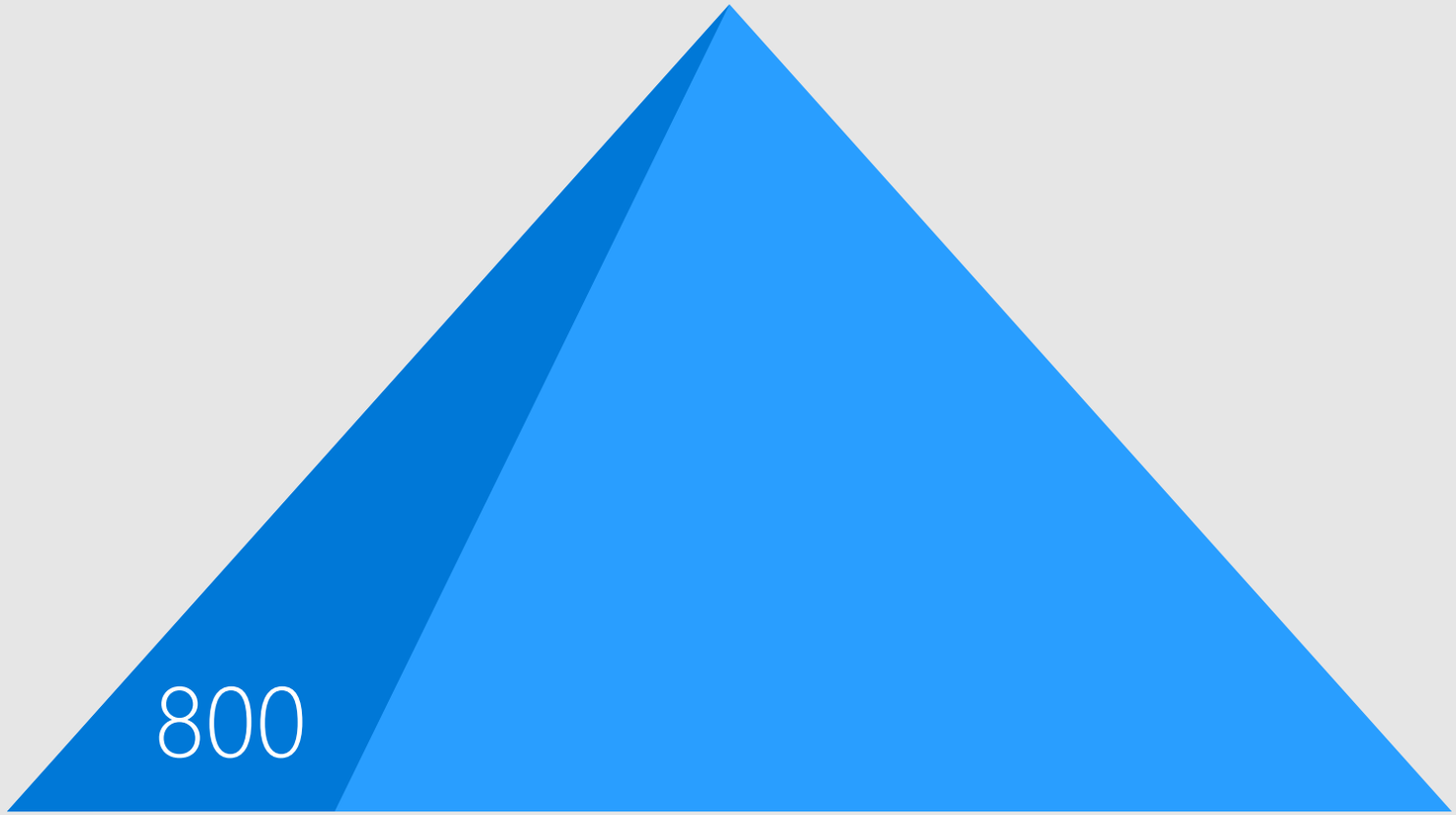




@AnthonyBorton



The Developer Division at Microsoft



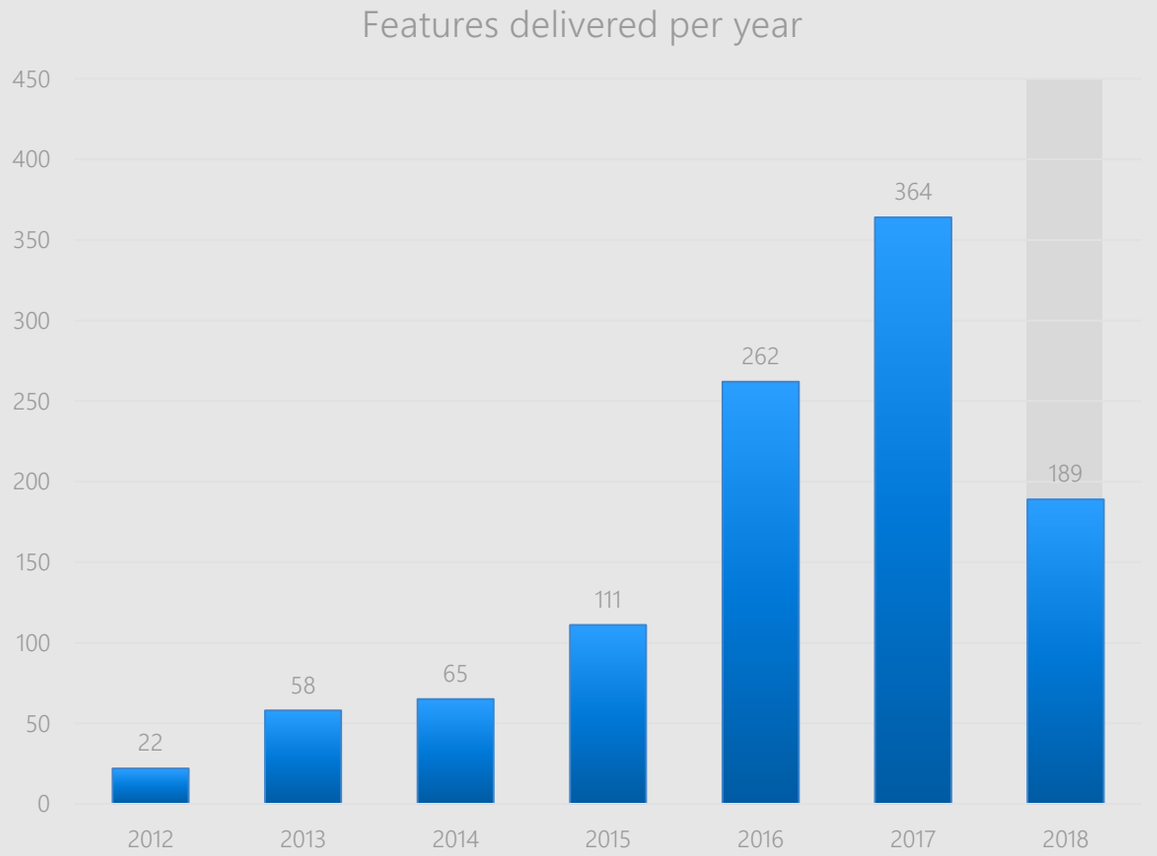
The Azure DevOps team… spread out across up to 40 feature teams

Azure **DevOps**

Features Delivered per Year

We are delivering value to customers and an increased velocity.

- More features in 2016 (262) than the previous 4 years combined (256 features).
- 364 features in 2017!



https://www.visualstudio.com/en-us/articles/news/features-timeline

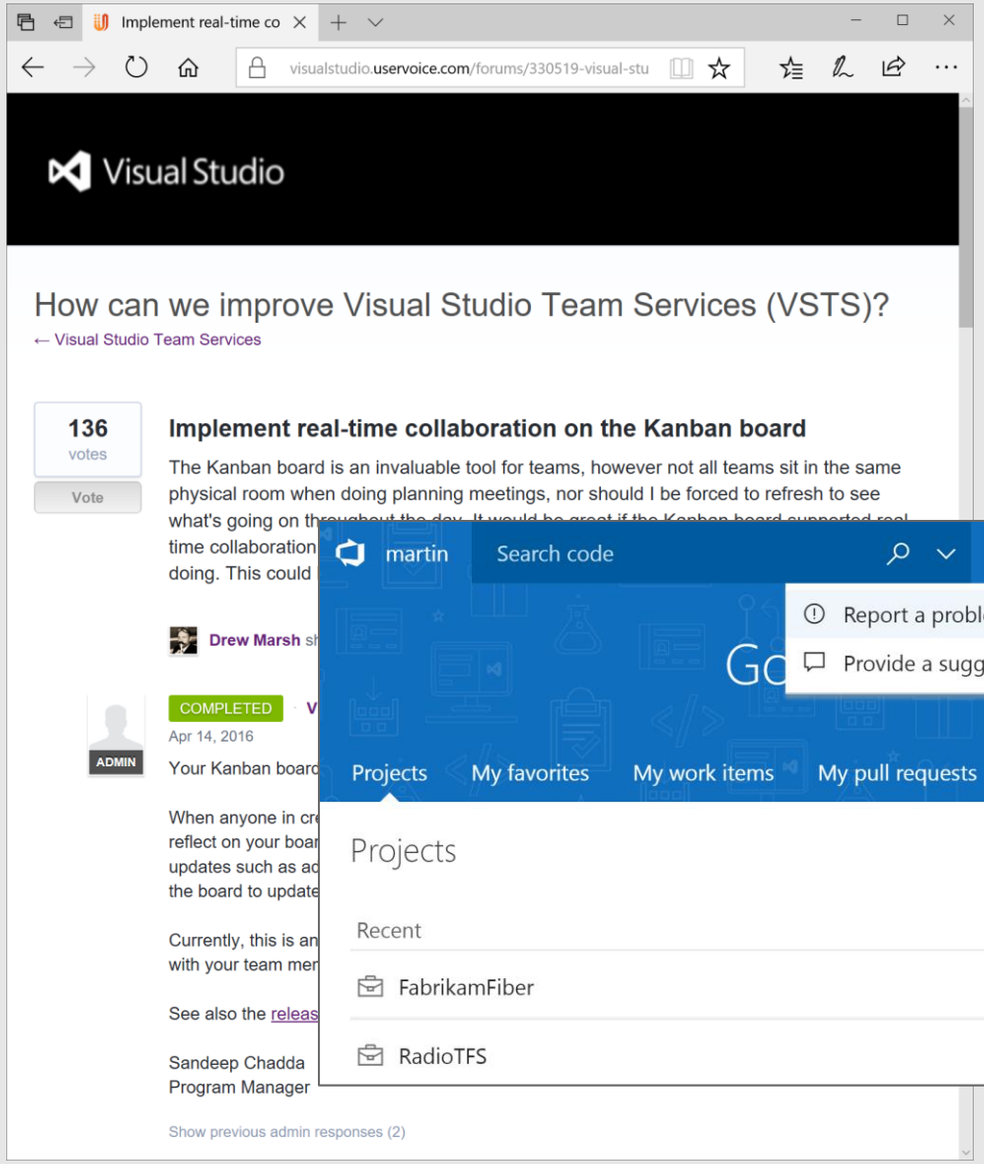
Four lessons we've learned so far

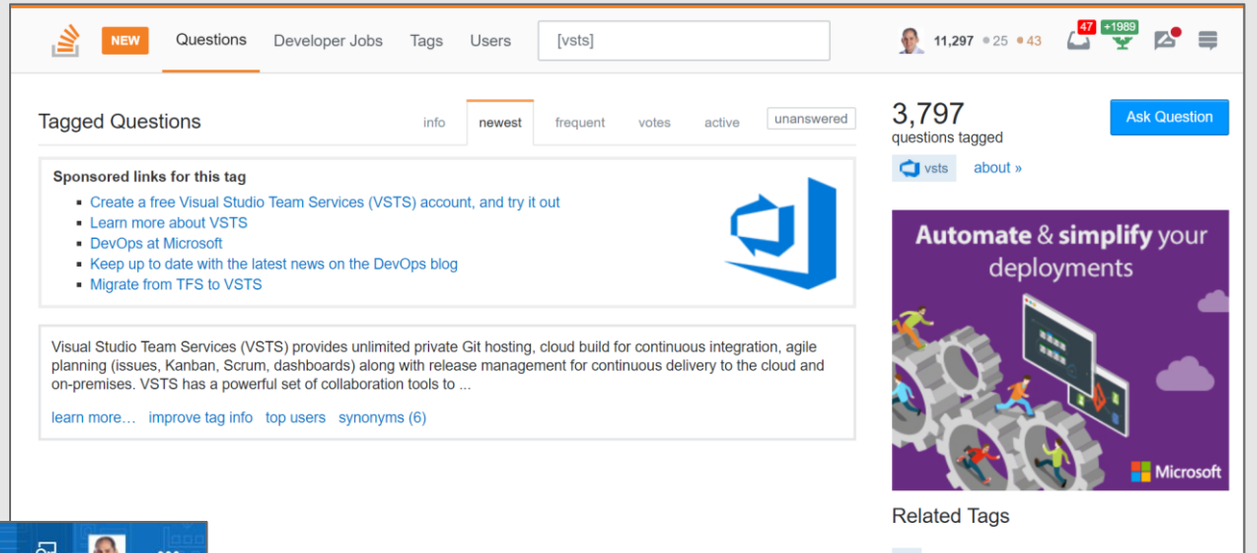
- Customer Focused
- Team Autonomy + Enterprise Alignment
- Shift Left Quality
- Safe Deployment and controlling exposure

Four lessons we've learned so far

- Customer Focused
- Team Autonomy + Enterprise Alignment
- Shift Left Quality
- Safe Deployment and controlling exposure

Listen to our customers





ation	Business	Champ	Engaged Users	Monthly Δ	Command Count	Availability	Import?	NPS
land	Professional services	aaronha	4468	3188	37,962,631	99.78%	yes	10
n, NL	Oil & gas	jeffbe	2144	101	81,118,987	99.82%	yes	9
NY	Financial information / analytics	amitgup	1897	42	65,377,799	99.87%	yes	10
C	Computer assisted legal research	roferg	1794	-8	23,119,242	99.97%	no	8
ngland	Professional services	midenn	1578	720	19,035,301	99.88%	yes	10
ermany	Reinsurance	ehofman	1204	27	37,827,300	99.79%	yes	9
A	Freight forwarding service	midenn	1187	6	64,852,531	99.92%	no	9
NY	Professional services	daazose	1165	244	46,686,753	99.84%	no	9
L, Houston	Oilfield services	samgu	1026	436	98,430,316	99.95%	no	7.5
e, AR	Retail	buckh	935	118	16,410,929	99.75%	yes	10
, SC	Software publishing	chrispat	927	96	28,950,208	99.88%	yes	9
ity, UT	IT Support Services	gauravsi	828	82	42,601,059	99.90%	yes	
	Heavy equipment manufacturing	abarr	746	117	6,893,030	99.91%	no	8
CA	Computer hardware & software	mariorod	742	-36	7,913,156	99.87%	no	8
ngland	Retail clothing	angelpe	706	80	76,135,638	99.44%	no	
ka, MN	Agriculture	taylaf	575	-1	12,056,989	99.61%	yes	10
Seattle, WA	IT Consulting	aaronha	574	119	5,621,066	99.95%	no	
Scottsdale, AZ	Project administration software	tarekm	556	-7	44,387,813	99.86%	yes	
Liberty Lake, WA	Energy and water resource mgmt	buckh	556	49	14,421,983	99.85%	no	8

What is DevOps?

DevOps is the union of people, process, and tools to enable continuous delivery of value to our customers.

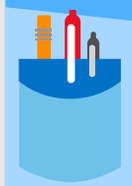
Increase flow of value

Shorten cycle times

Continuously Improve

Build-Measure-Learn

Hypothesis



We believe {customer segment} wants {product/feature} because {value prop}

Experiment

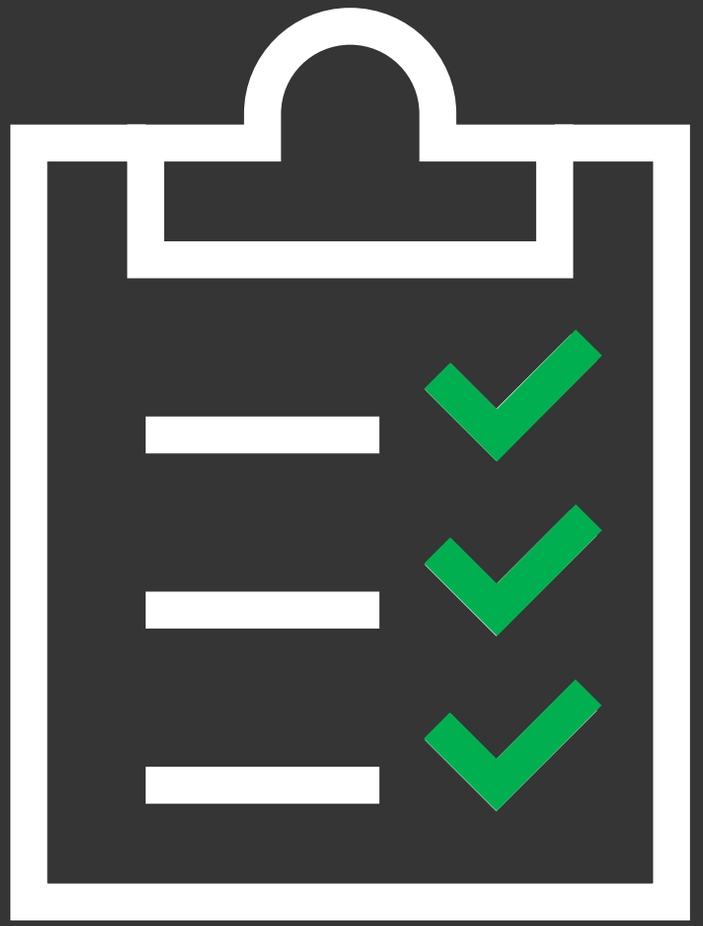
To prove or disprove the above, the team will conduct the following experiment(s): …

Learning

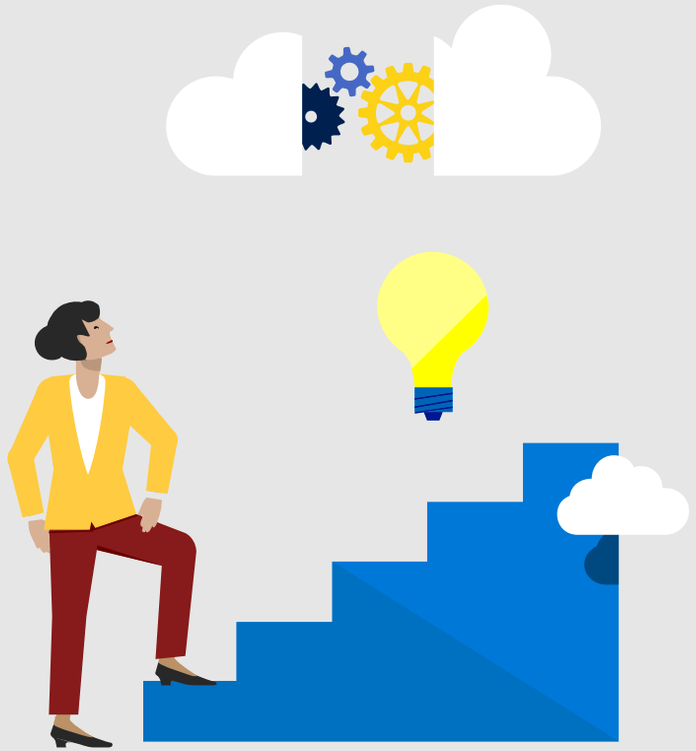
The above experiment(s) proved or disproved the hypothesis by impacting the following metric(s): …

Our Definition of Done

Live in production, collecting telemetry that examines the hypothesis which motivated the deployment.



Four lessons we've learned so far



- Customer Focused
- **Team Autonomy + Enterprise Alignment**
- Shift Left Quality
- Safe Deployment and controlling exposure

Microsof

Enc

- B
- E
- W
- H

WIKIPEDIA
The Free Encyclopedia

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes

Article | Talk

Encarta

From Wikipedia, the free encyclopedia

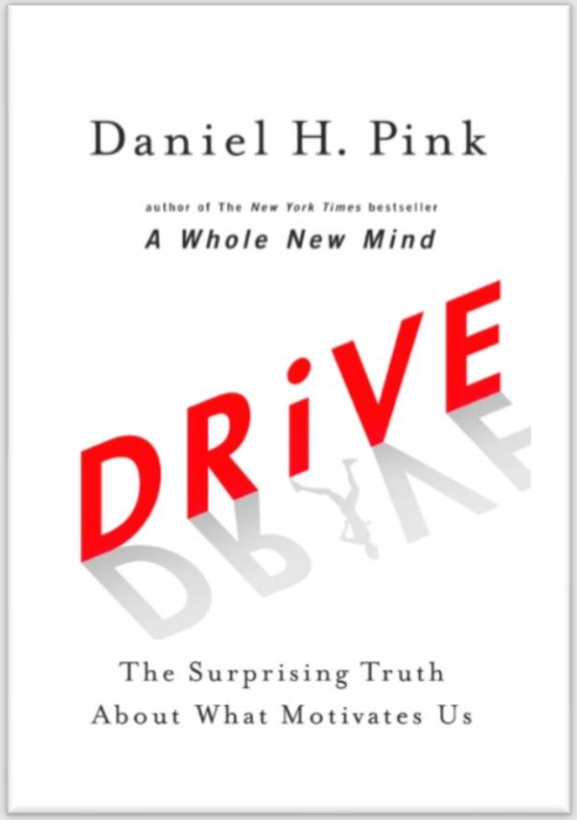
Microsoft Encarta was a digital multimedia encyclopedia that was published by Microsoft Corporation from 1993 to 2009. In 2008, the complete English version, *Encarta Premium*, consisted of more than 62,000 articles,[1] numerous photos and illustrations, music clips, videos, interactive contents, timelines, maps, atlases and homework tools. It was available on the World Wide Web by annual subscription or by purchase on DVD or multiple CDs. Many articles could also be viewed free online with advertisements.[2]

Microsoft published similar encyclopedias under the *Encarta* trademark in various languages, including German, French, Spanish, Dutch, Italian, Portuguese and Japanese. Localized versions contained contents licensed from national sources and more or less content than the full English version. For example, the Dutch version had content from the Dutch *Winkler Prins* encyclopedia.

e
me
ant to

Agile at Scale with Aligned Autonomy

"Let's try to give our teams three things….
Autonomy, Mastery, Purpose"



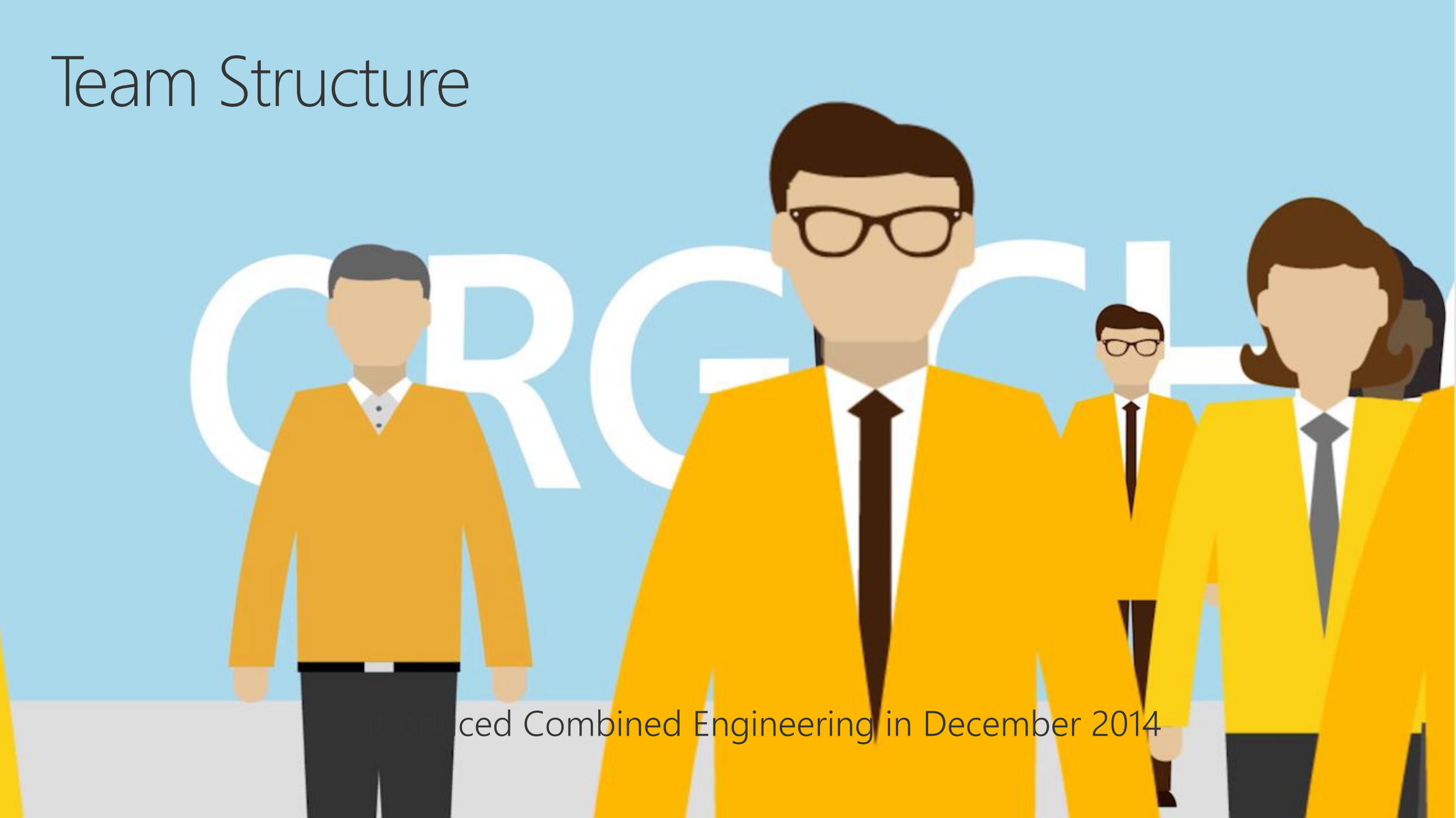
- Organisation
- Roles
- Teams
- Cadence
- Taxonomy

Alignment

- Plan
- Practices

Autonomy

Team Structure



Introduced Combined Engineering in December 2014

ORG CHART

PROGRAM MANAGEMENT

DEVELOPMENT

TESTING

ORG CHART

PROGRAM MANAGEMENT

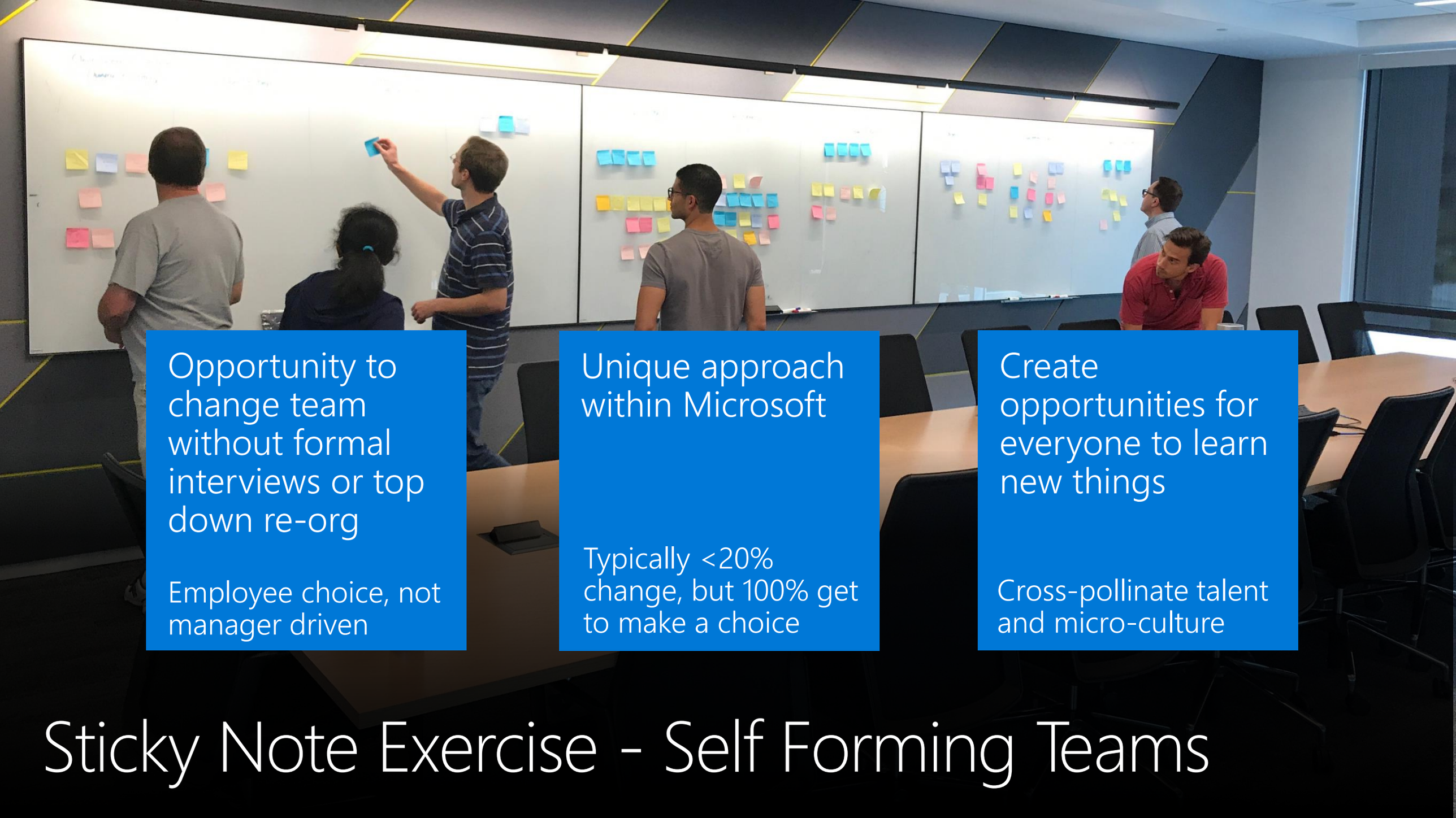
ENGINEERING

OPs

Teams

- Physical team rooms
- Cross discipline
- 10-12 people
- Self managing
- Clear charter and goals
- Intact for 12-18 months

- Own features in production
- Own deployment of features



Opportunity to change team without formal interviews or top down re-org

Employee choice, not manager driven

Unique approach within Microsoft

Typically <20% change, but 100% get to make a choice

Create opportunities for everyone to learn new things

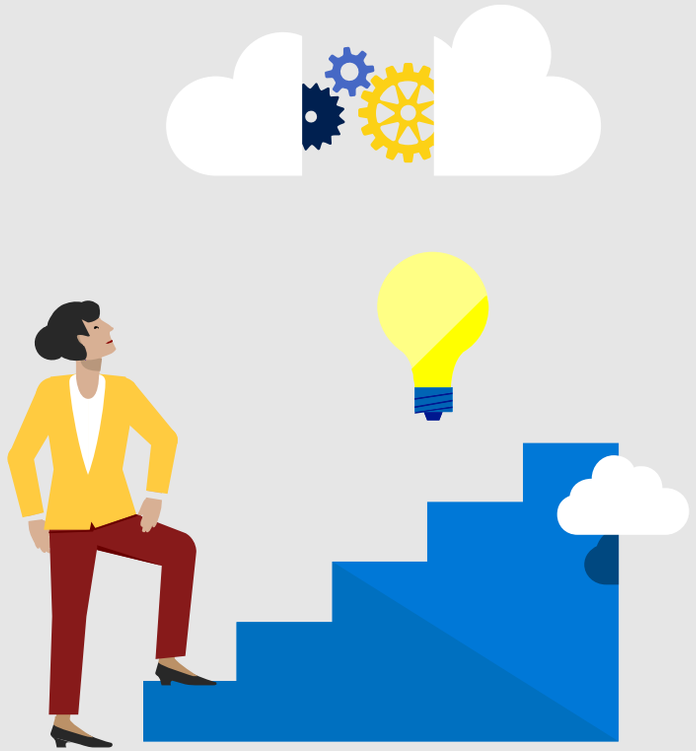
Cross-pollinate talent and micro-culture

Sticky Note Exercise - Self Forming Teams

Transformation Benefits

- Teams feel that they own the customer experience & are responsible for improving it
- Teams are continually planning
- Planning is driven by continual learning
  - Telemetry on usage
  - Customer feedback
  - "Failing fast" through in incremental execution and delivery
- Opportunities to continually evaluate progress
- We can react… *if & when* we need to change course

Four lessons we've learned so far



- Customer Focused
- Team Autonomy + Enterprise Alignment
- **Shift Left Quality**
- Safe Deployment and controlling exposure

Testing circa 2010 – arrival of the Cloud Services

New constraints and requirements

- Faster cadence, even faster cadence, and more
- Lack of customer validation through Beta, RC etc.
- Micro-services deployed independently
- High availability, no downtime deployments
- ….

Initial response and approach

- Do the traditional waterfall dev/test model but faster
- Pushed for faster automation
- Test Selection techniques as a way of survival

Our problems: September 2014

Tests took too long

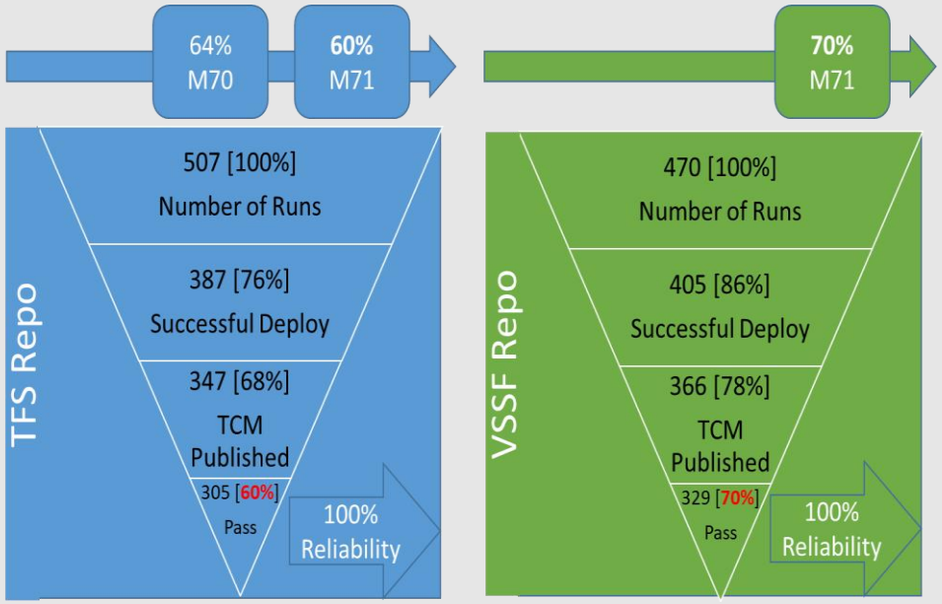
- Over 22 hours for nightly run
- 2 days for the full run

Tests failed frequently

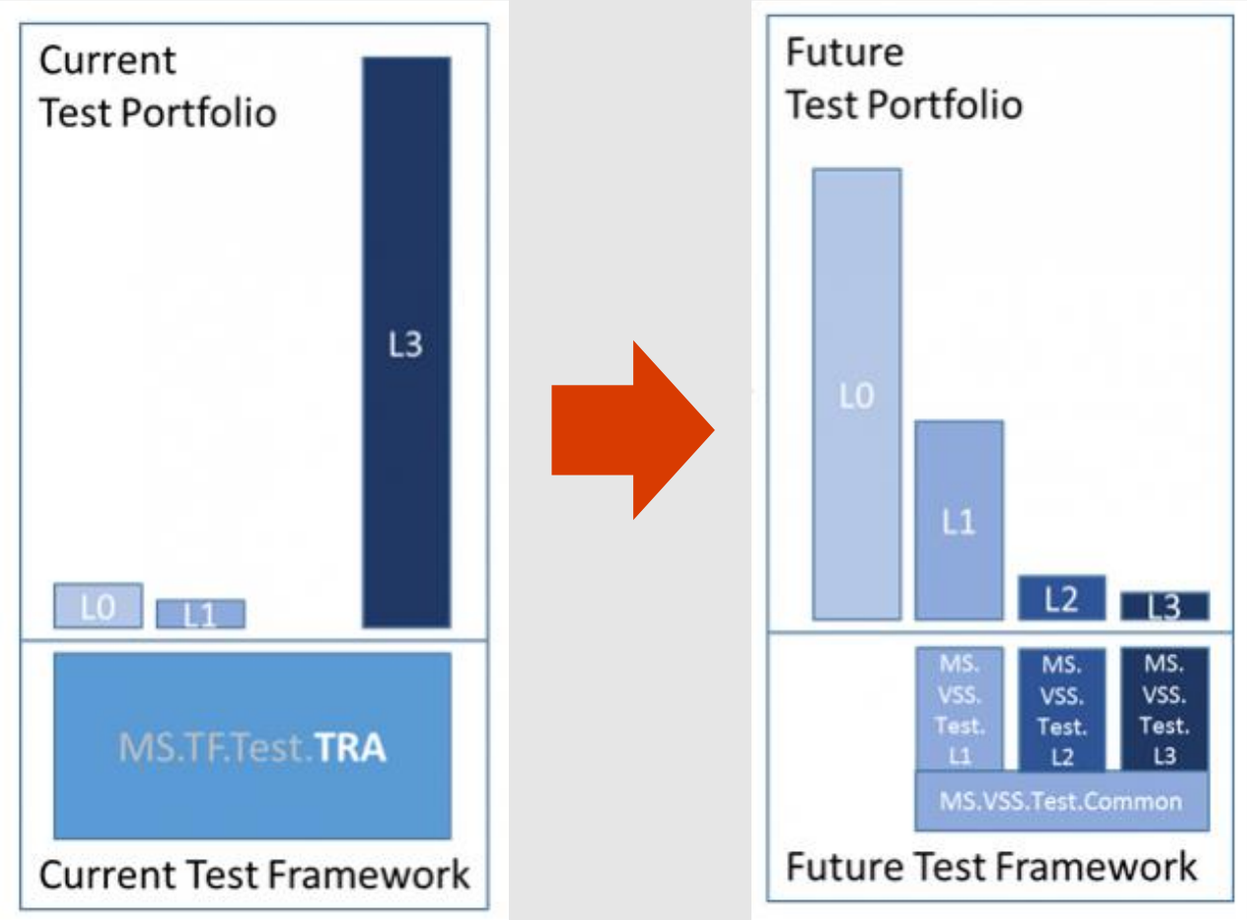
- Only ~60% of P0 runs passed 100%;
  Each NAR suite had many failures

Quality signal unreliable in Master

- Test failure analysis was too costly



Published VSTS Quality Vision : Feb ‘15



Principles

- Tests should be written at the lowest level possible
- Write once, run anywhere including production system
- Product is designed for testability
- Test code is product code, only reliable tests survive

Test Taxonomy

Levels can roughly be understood as a measure of external dependencies

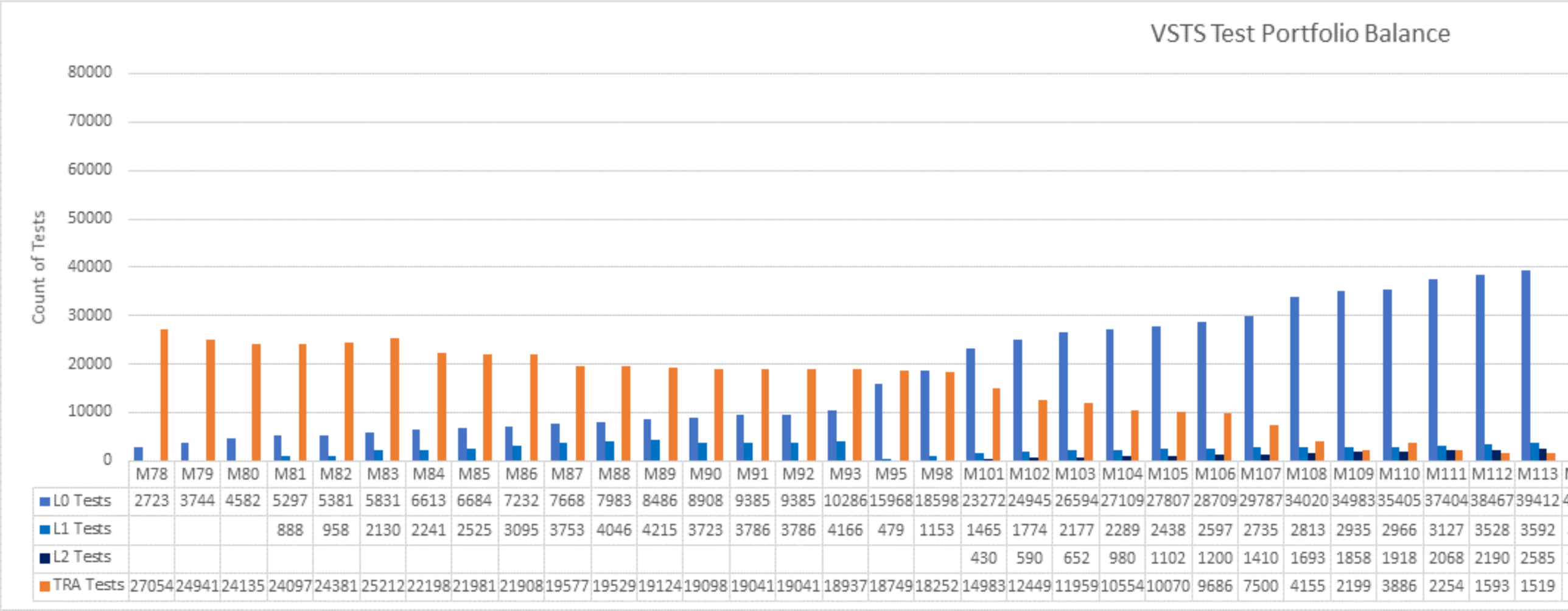
- Unit Tests

L0	Broad class of fast in-memory unit tests
L1	Unit tests with more complex requirements e.g. SQL

- Functional Tests

L2	Functional tests run against “testable” service deployment
L3	Restricted class integration tests that run against production

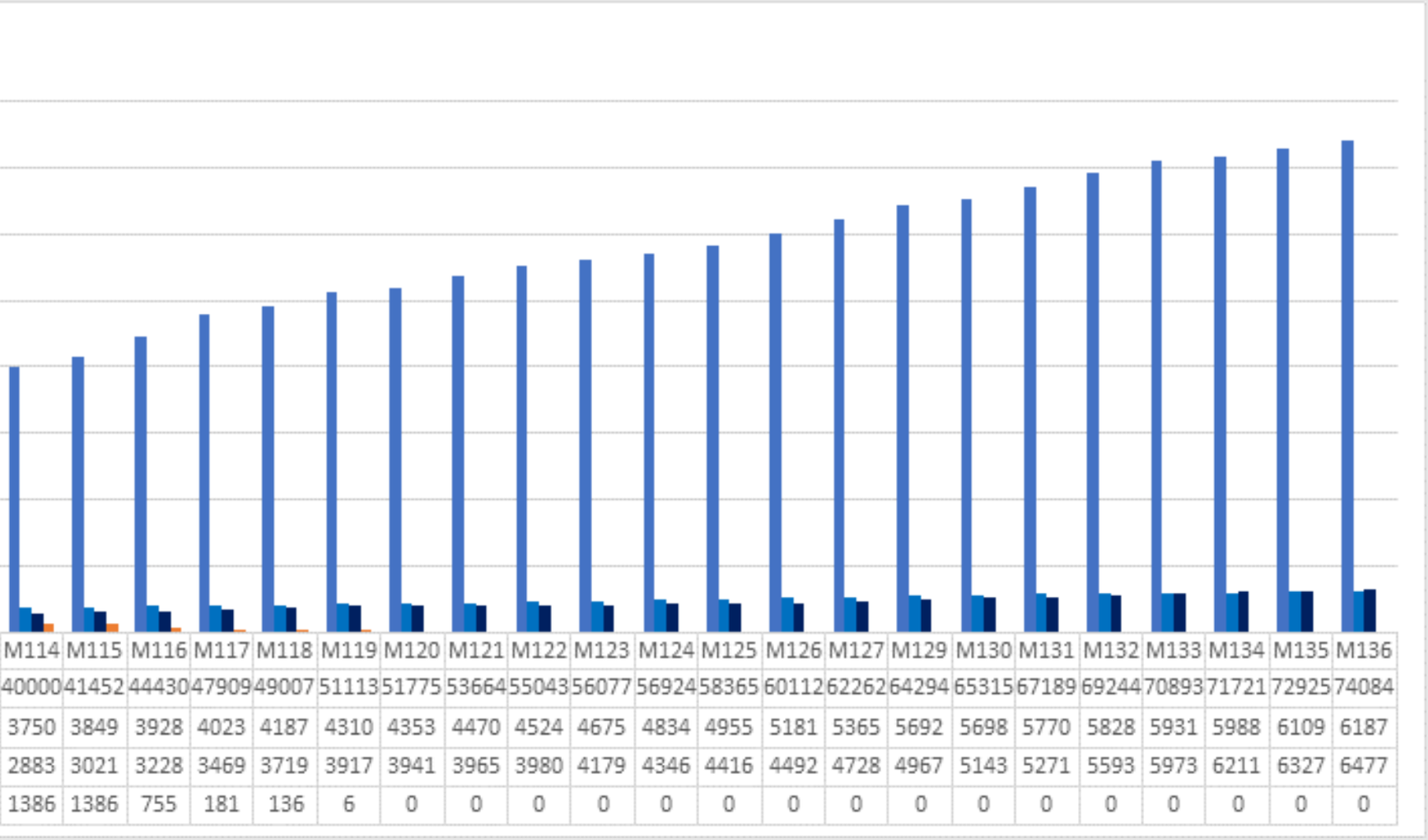
Test portfolio over time



	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
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
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
L2 Tests																			430	590	652	980	1102	1200	1410	1693	1858	1918	2068	2190	2585
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

@AnthonyBorton

Test portfolio over time



M114	M115	M116	M117	M118	M119	M120	M121	M122	M123	M124	M125	M126	M127	M129	M130	M131	M132	M133	M134	M135	M136
40000	41452	44430	47909	49007	51113	51775	53664	55043	56077	56924	58365	60112	62262	64294	65315	67189	69244	70893	71721	72925	74084
3750	3849	3928	4023	4187	4310	4353	4470	4524	4675	4834	4955	5181	5365	5692	5698	5770	5828	5931	5988	6109	6187
2883	3021	3228	3469	3719	3917	3941	3965	3980	4179	4346	4416	4492	4728	4967	5143	5271	5593	5973	6211	6327	6477
1386	1386	755	181	136	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TYPE	M78	M136	DELTA
L0	2723	74084	+ 71,361
L1		6187	+ 6,187
L2		6477	+ 6,477
TRA	27054	0	- 27,054

@AnthonyBorton

Four lessons we've learned so far

- Customer Focused
- Team Autonomy + Enterprise Alignment
- Shift Left Quality
- **Safe Deployment and controlling exposure**

Early Principles

The same tools we use to deploy to production we use in dev and test environments

The quality signals we look at to green light deployments are tracked constantly every day

Deployments take zero down time

Deployments happen during working hours

What do feature flags give us?

Decouple deployment and exposure

Flags provide runtime control down to individual user

Change without redeployment

Controlled via PowerShell or web UI

Support early feedback, experimentation

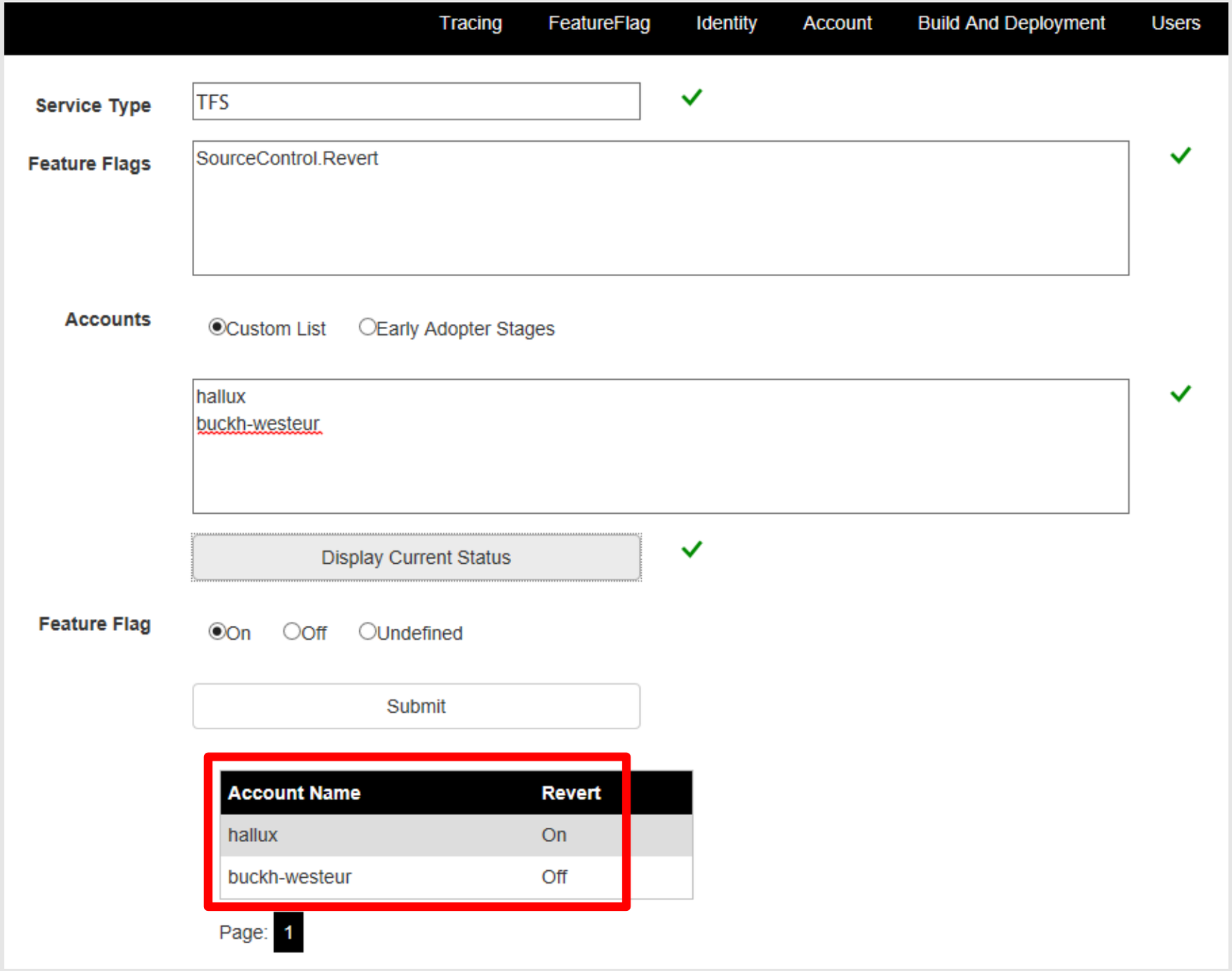
Quick off switch

Control

PowerShell

- Get-FeatureFlag
- Set-FeatureFlag

Web UI

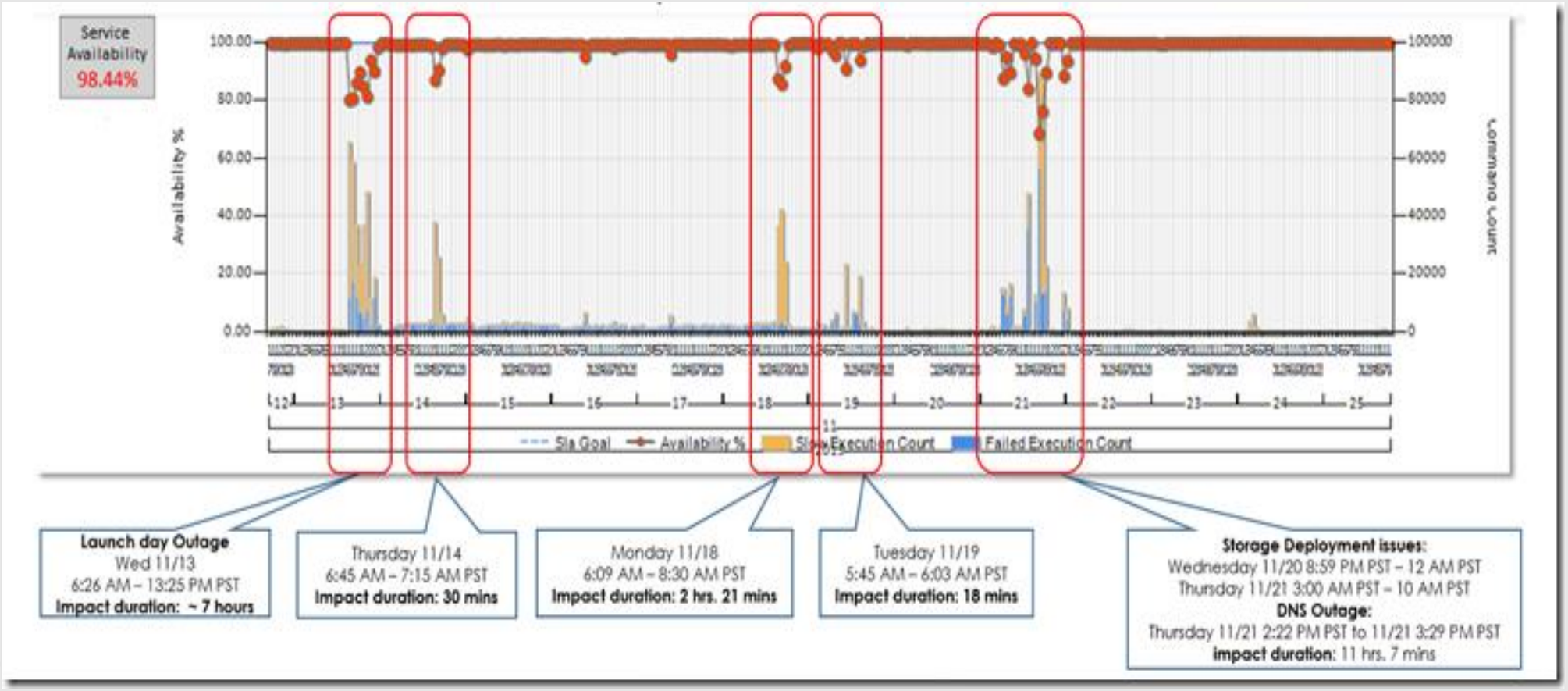


That's great…what could go wrong?

Features to be revealed at Connect 2013 event

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

It didn't go well.



@AnthonyB

What went wrong?

Turned on flags in production morning of event
On the only instance we had at the time

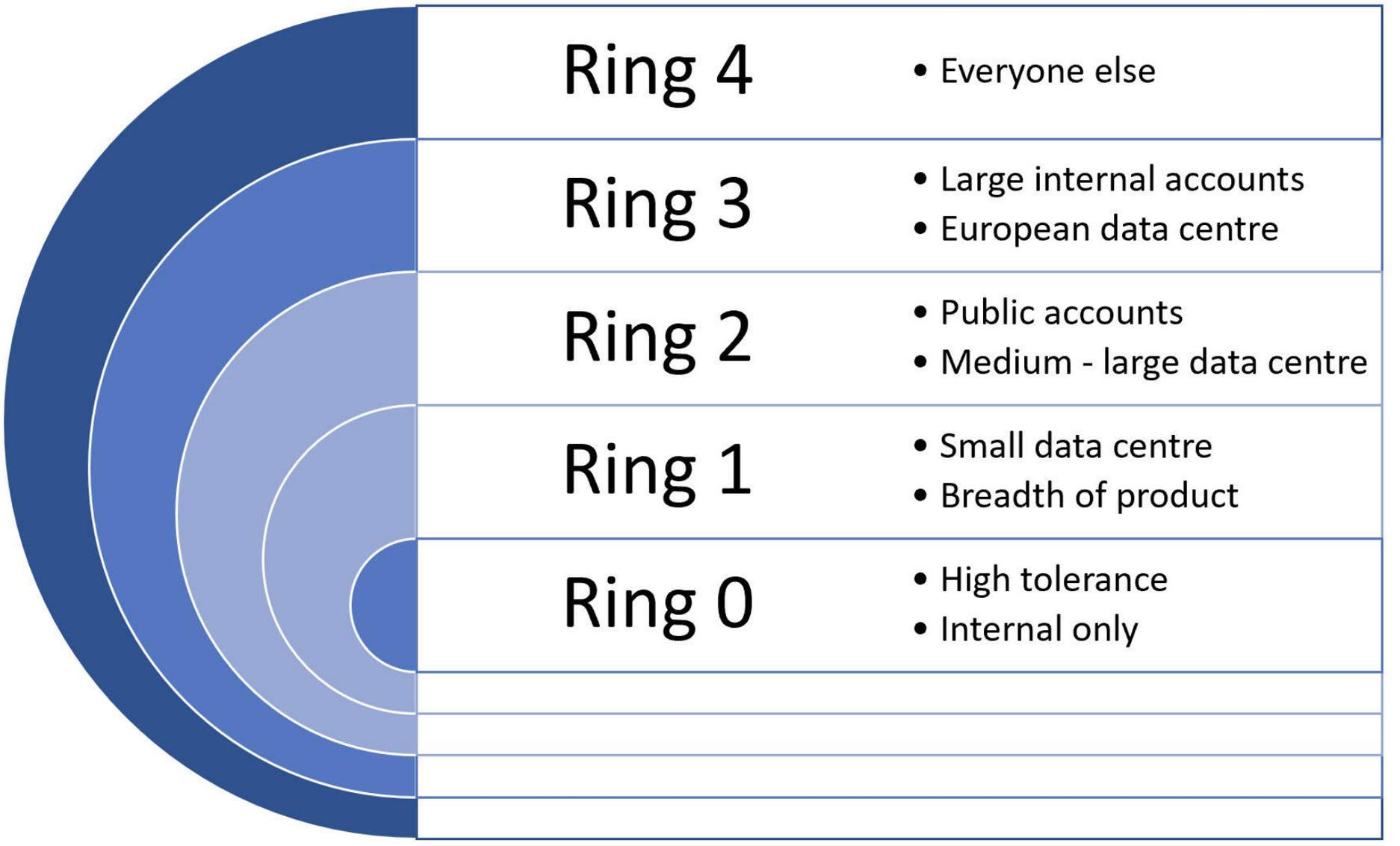
Now...
Incrementally turned on
Turned on completely at least 24 hours ahead of an event

What is Safe Deployment?

Deploy changes to risk tolerant customers first, progressively roll out to larger and larger sets of customers

Automated health checks and roll back

Deployment rings




The journey has its ups and down,
but in the end it’s worth it

Thank you


anthony.borton@microsoft.com


@AnthonyBorton


https://www.linkedin.com/in/anthonyborton/


https://anthonyborton.com

Microsoft