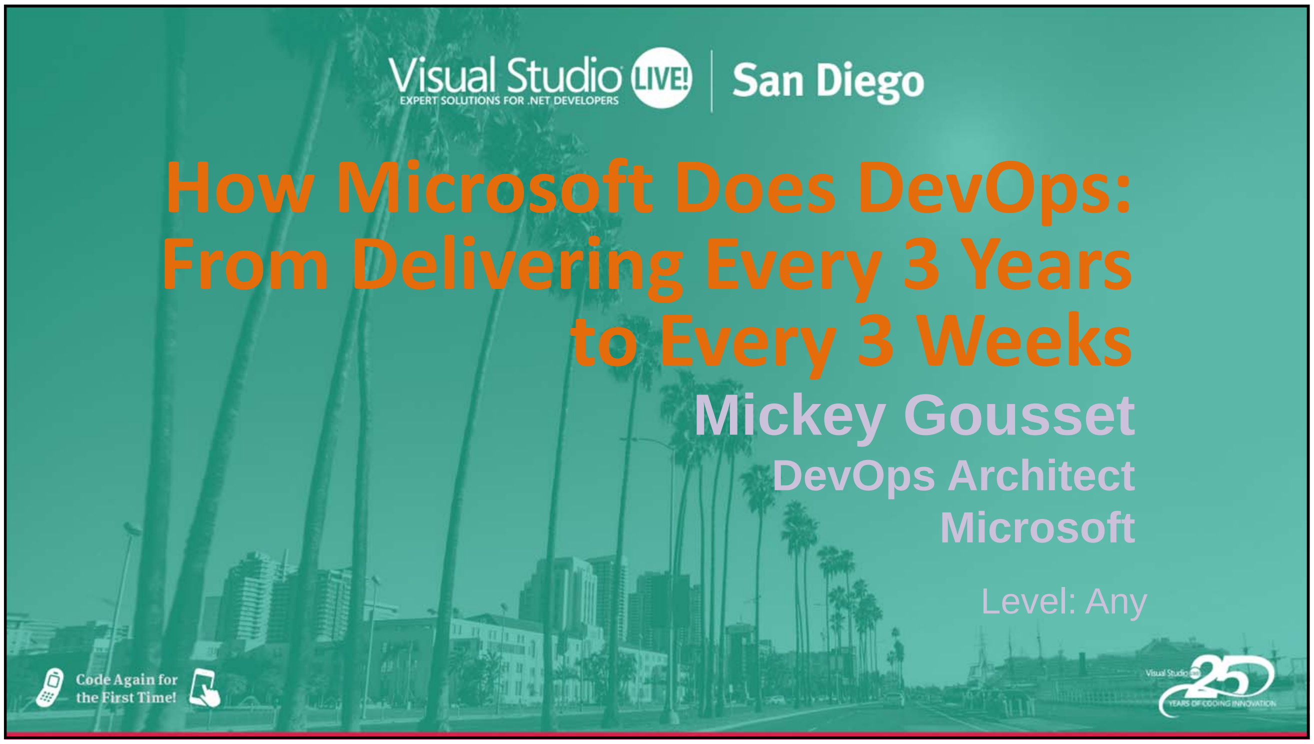


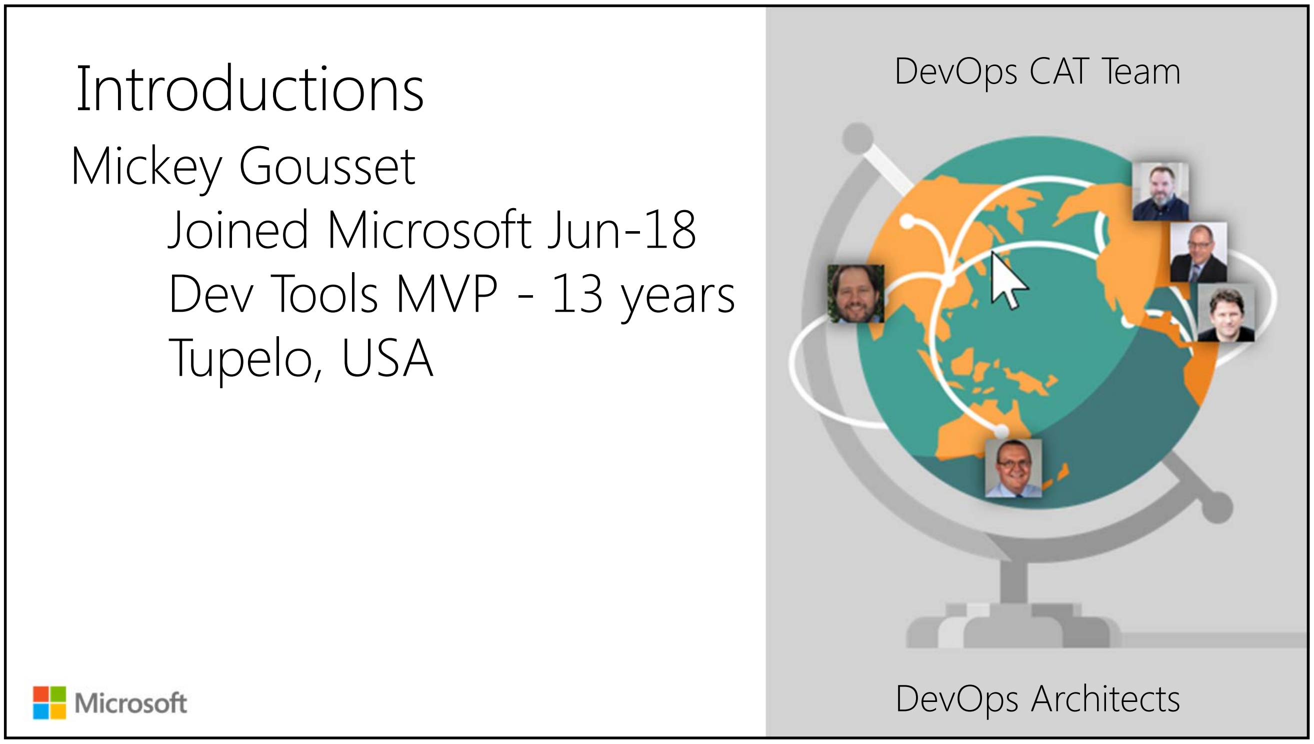
Visual Studio LIVE! EXPERT SOLUTIONS FOR .NET DEVELOPERS | San Diego

How Microsoft Does DevOps: From Delivering Every 3 Years to Every 3 Weeks

Mickey Gousset
DevOps Architect
Microsoft

Level: Any

Code Again for the First Time!
Visual Studio LIVE! 25 YEARS OF CODING INNOVATION



Introductions

Mickey Gousset

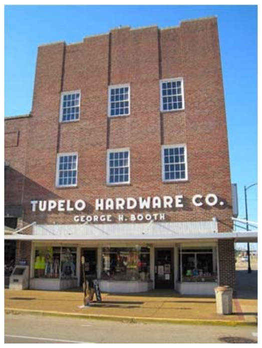
- Joined Microsoft Jun-18
- Dev Tools MVP - 13 years
- Tupelo, USA

DevOps CAT Team

DevOps Architects

Microsoft

What is Tupelo, MS world-famous for?



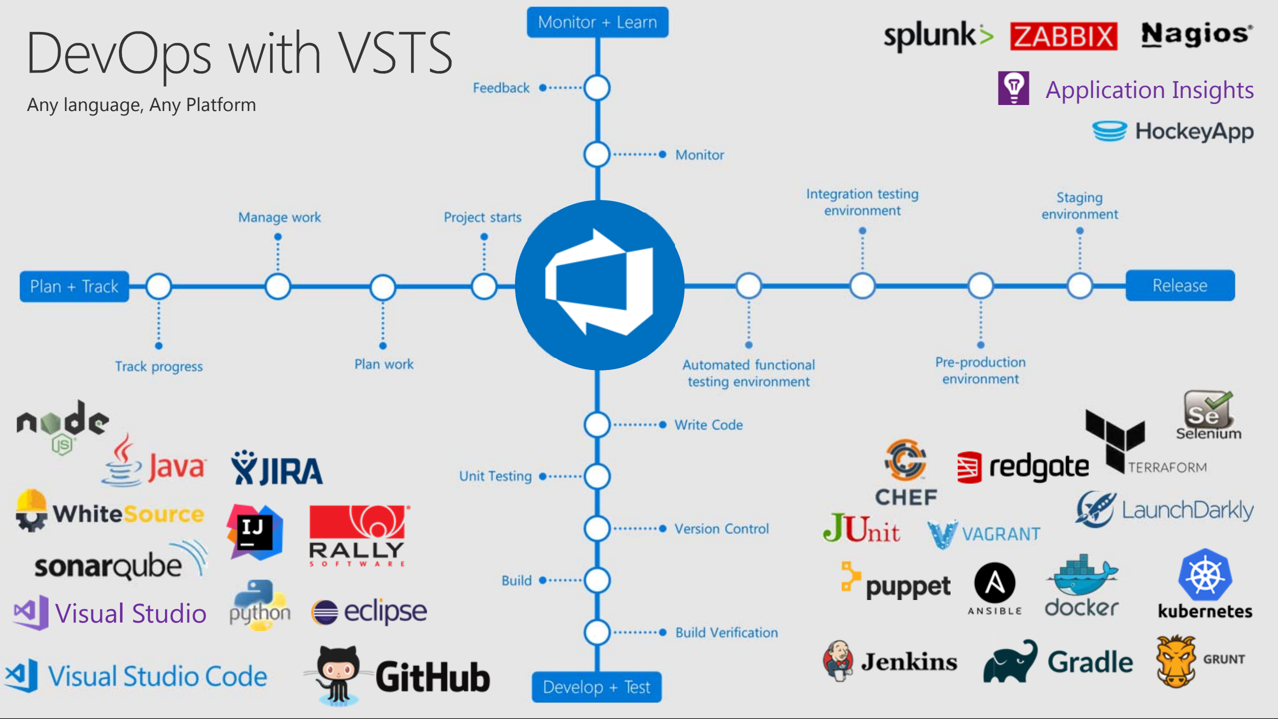
DevOps is the union of people, process, and products to enable continuous delivery of value to our end users.

- Donovan Brown

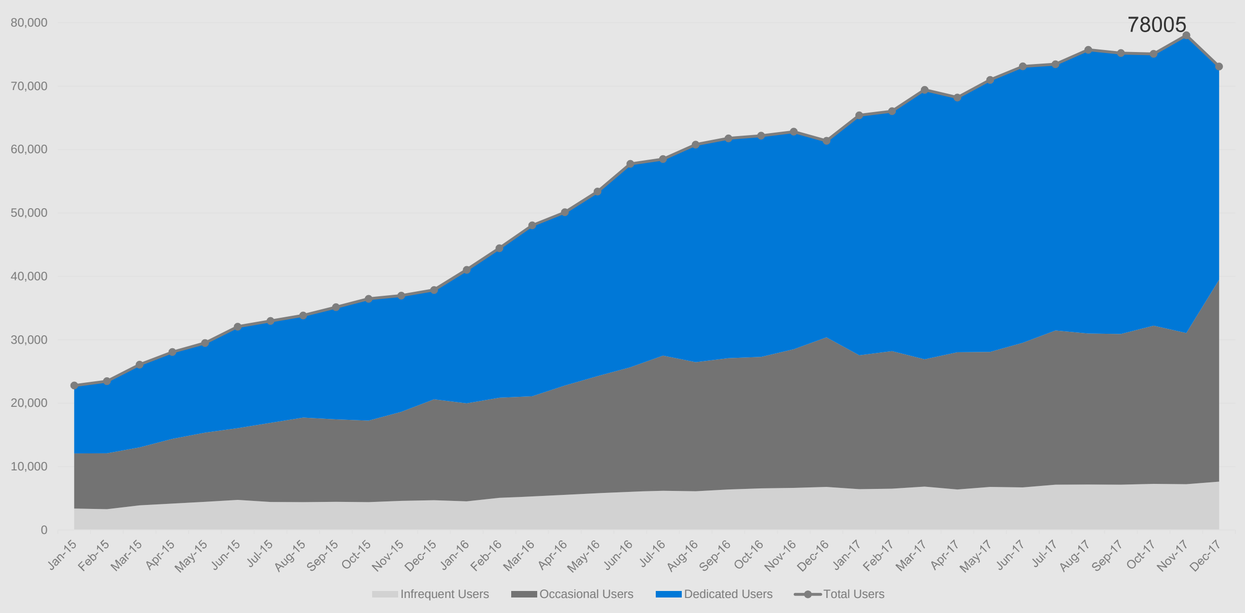
http://bit.ly/WhatIs-DevOps

DevOps with VSTS

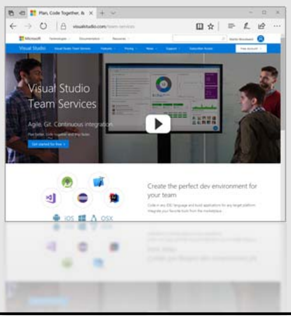
Any language, Any Platform

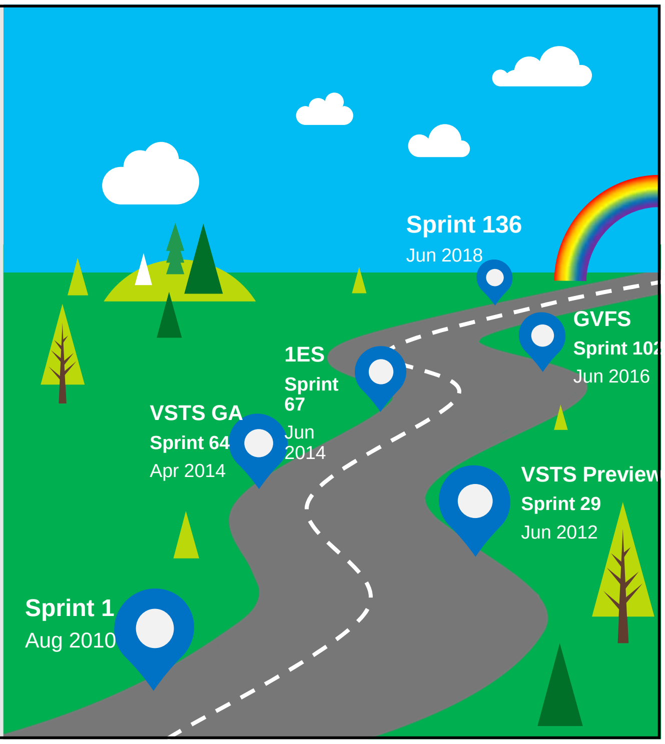


VSTS: Millions of users + Most of Microsoft



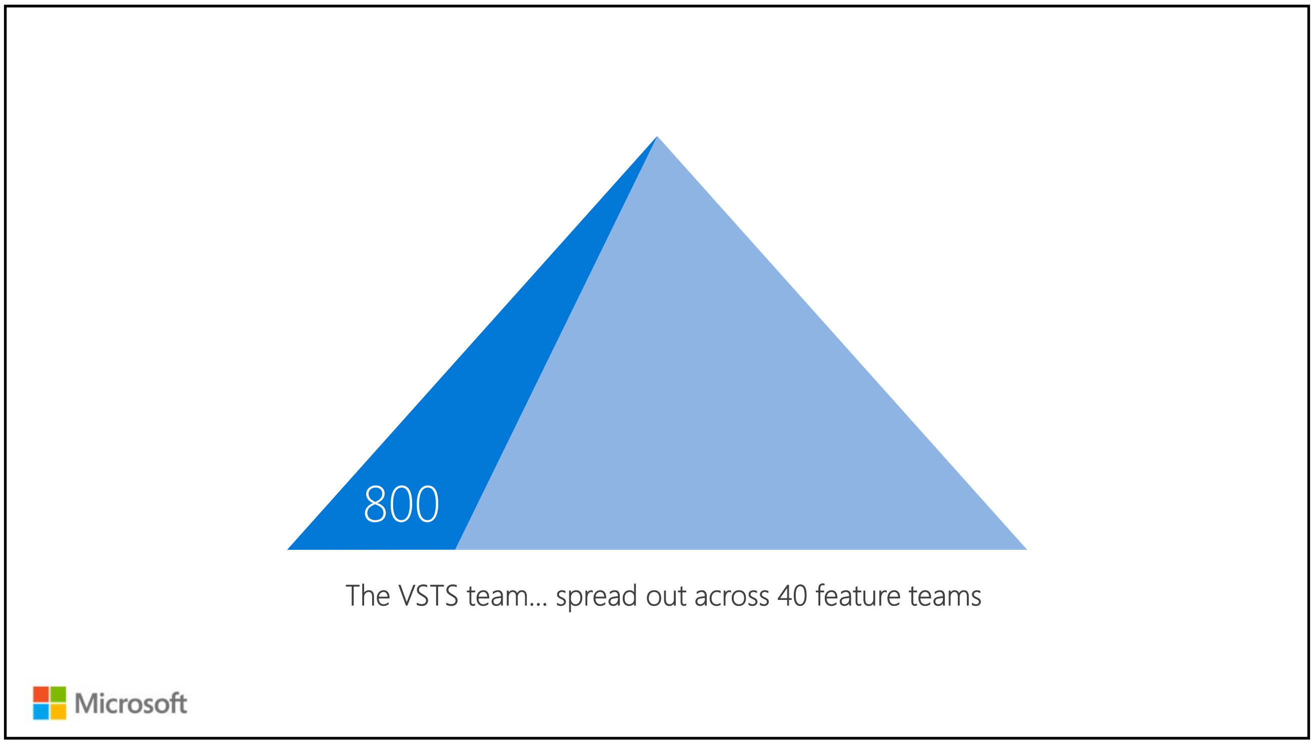
The Journey to DevOps





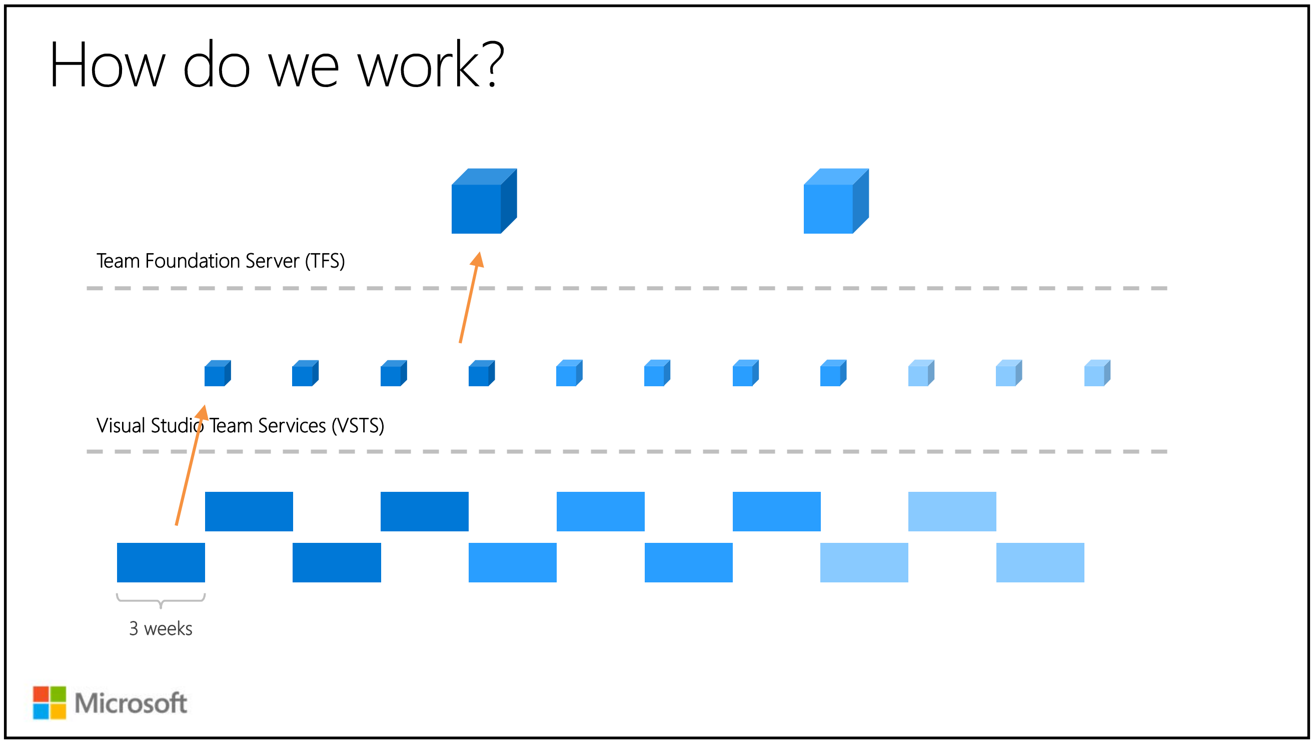
3,500

The Developer Division at Microsoft



The VSTS team… spread out across 40 feature teams

Microsoft



How do we work?

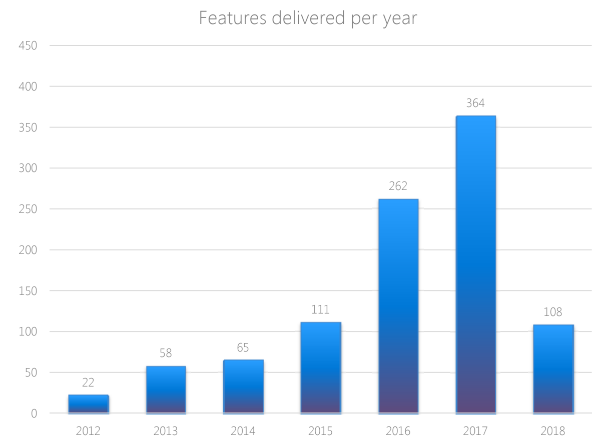
Team Foundation Server (TFS)
Visual Studio Team Services (VSTS)
3 weeks

Microsoft

Features Delivered per Year

We are delivering value to customers and an increased velocity.

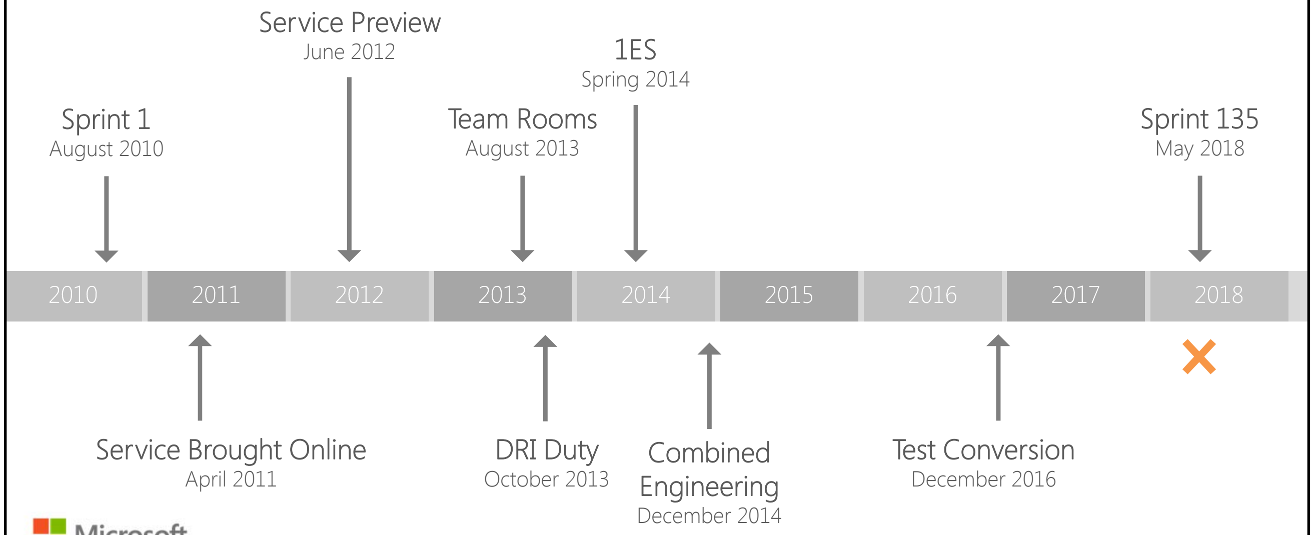
- More features in 2016 (262) than the previous 4 years combined (256 features).
- 364 features in 2017!



Microsoft

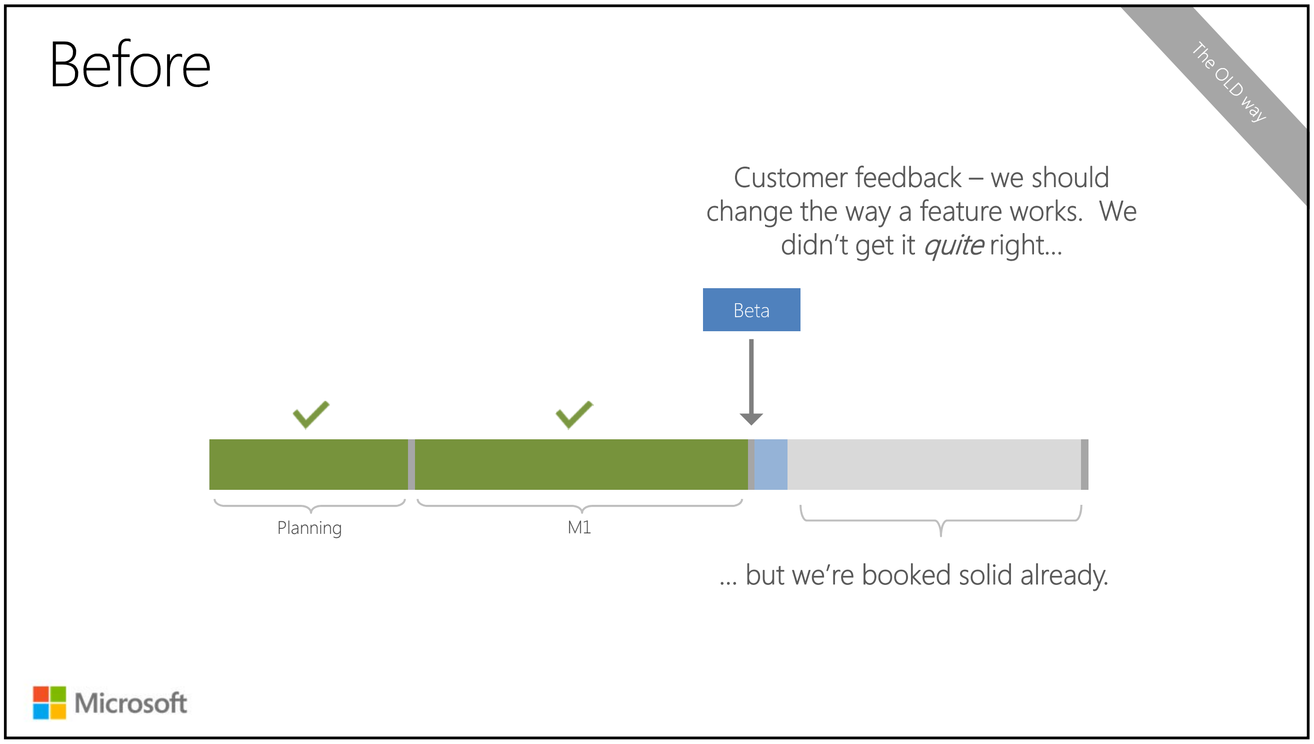
https://www.visualstudio.com/en-us/articles/news/features-timeline

The Journey



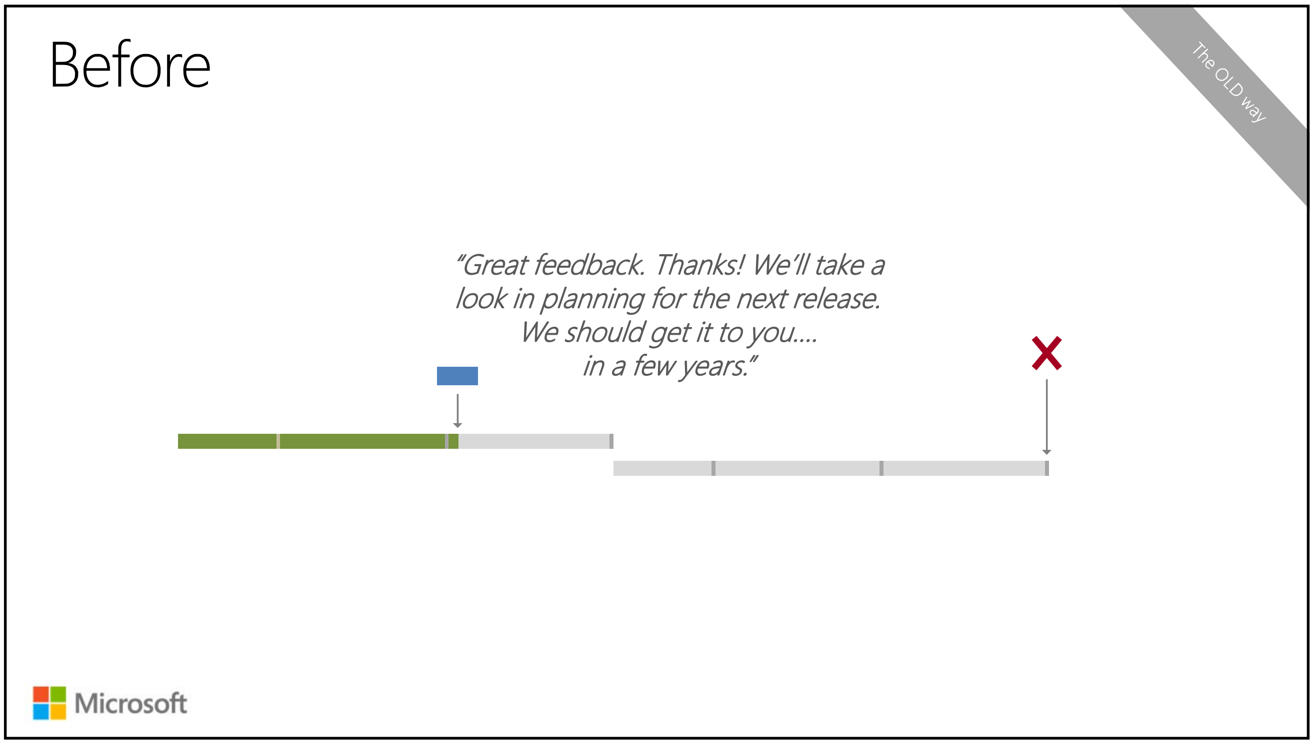
Microsoft

What did it look like before?



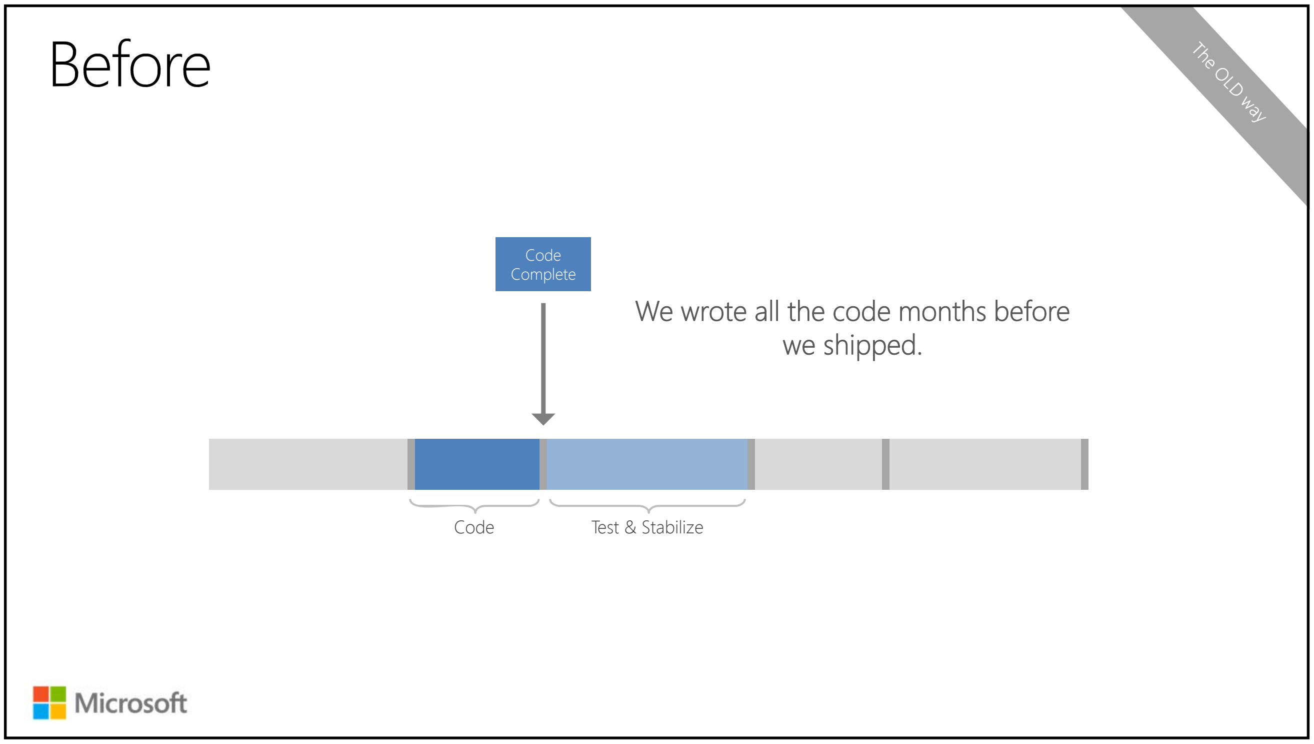
Before

The OLD way
Customer feedback – we should change the way a feature works. We didn't get it *quite* right…
Beta
✓
✓
Planning
M1
… but we're booked solid already.
Microsoft



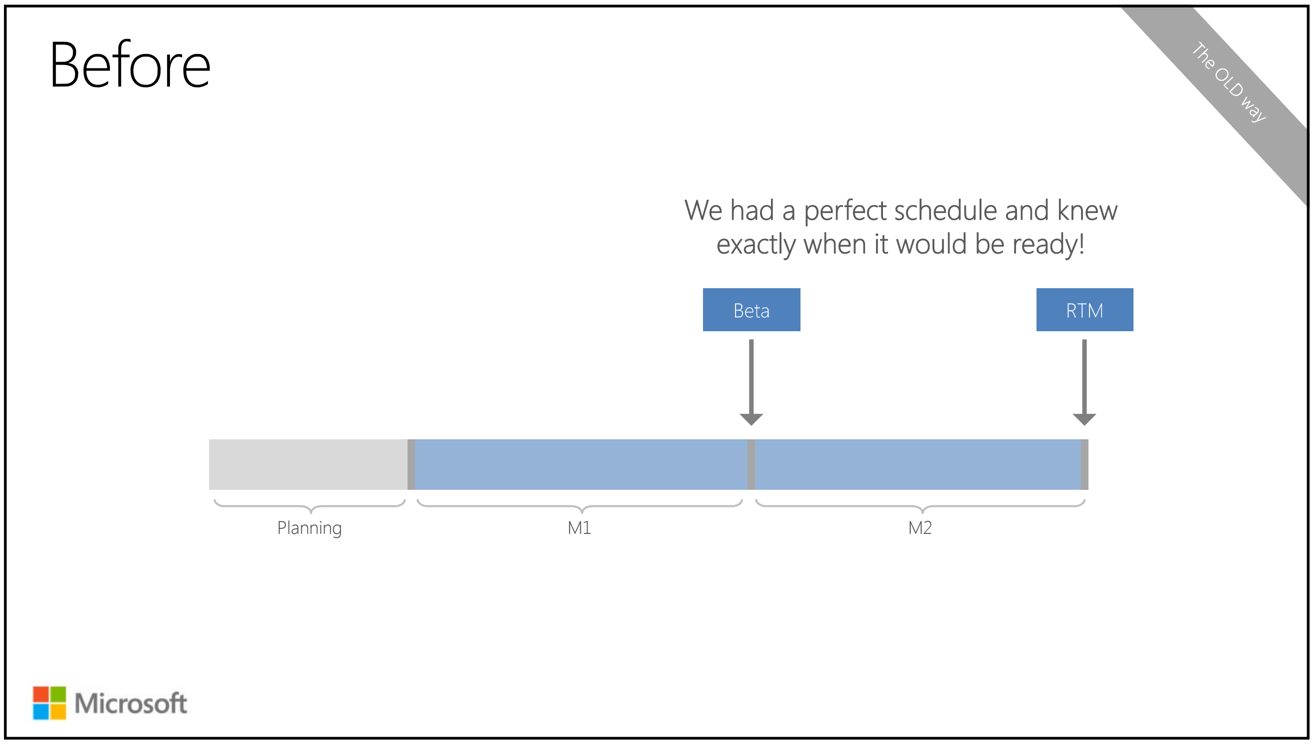
Before

"Great feedback. Thanks! We'll take a look in planning for the next release. We should get it to you…. in a few years."



Before

We wrote all the code months before we shipped.

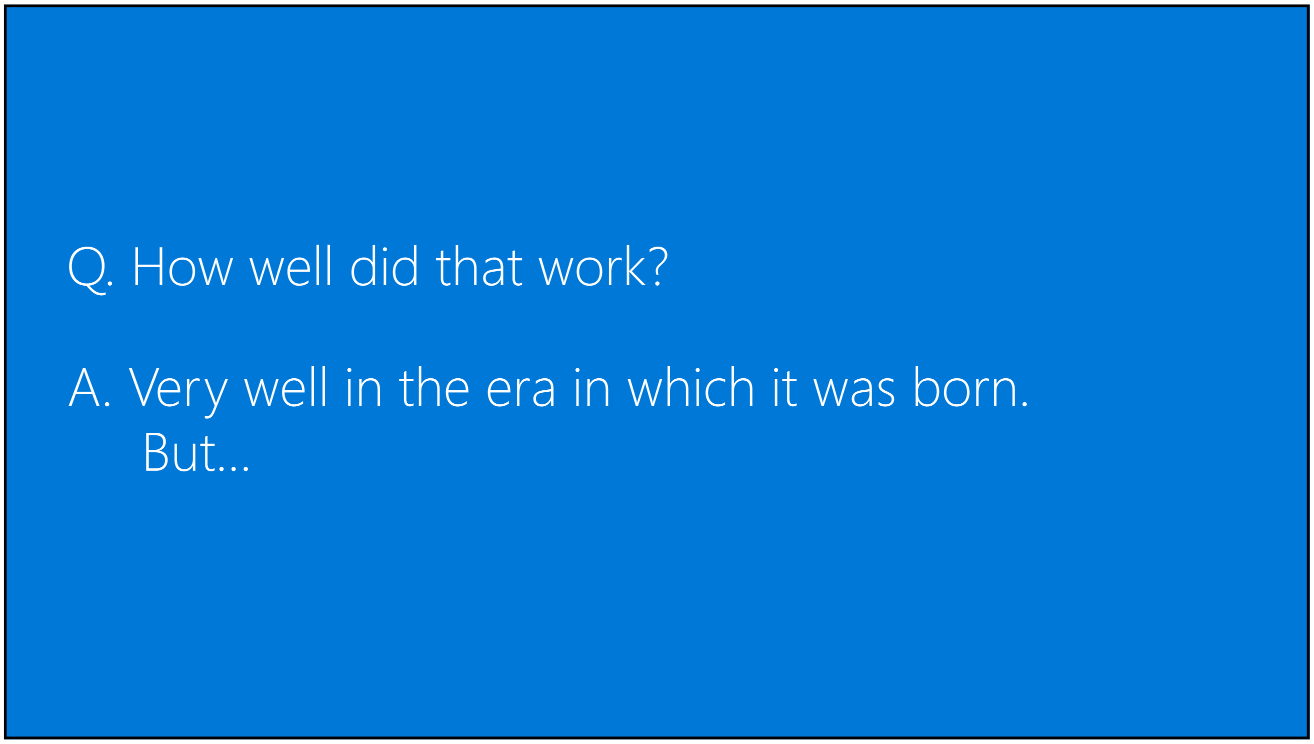


Before

The OLD way

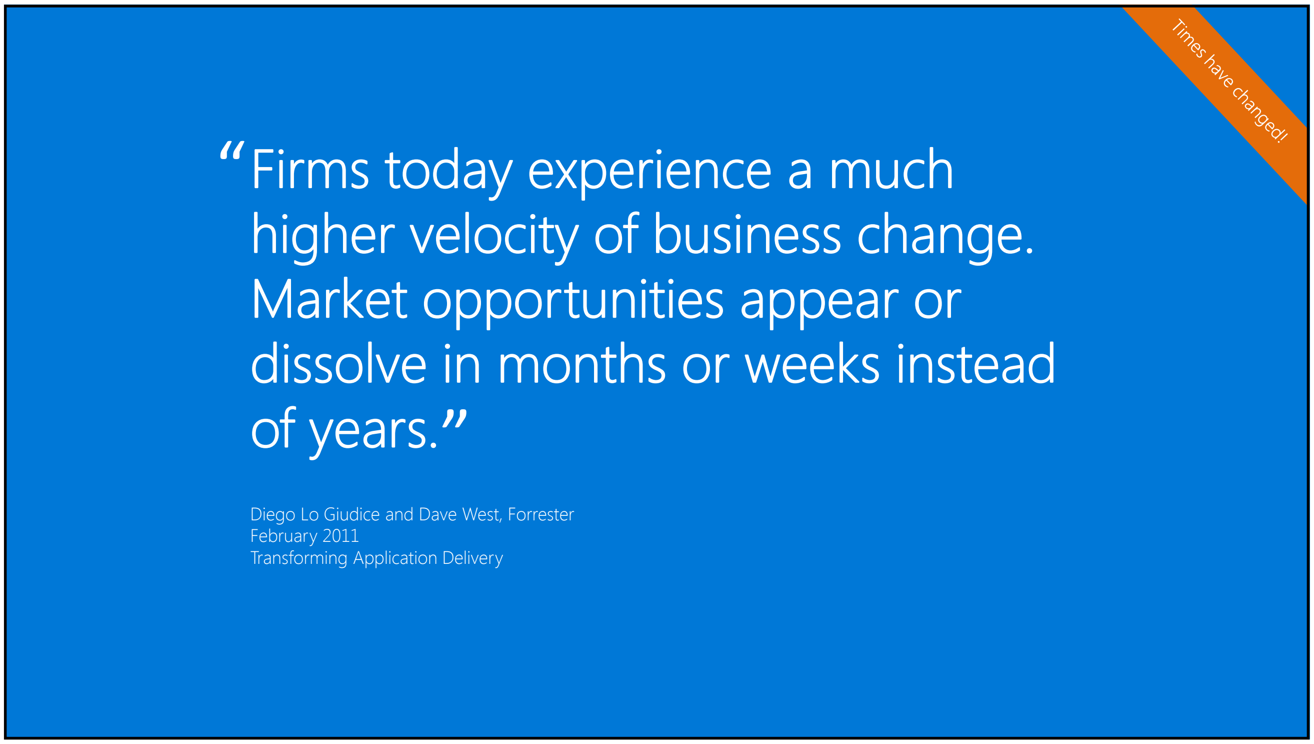
We had a perfect schedule and knew exactly when it would be ready!

Microsoft



Q. How well did that work?

A. Very well in the era in which it was born.
But…



Times have changed!

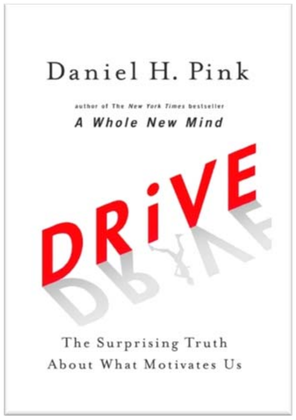
“Firms today experience a much higher velocity of business change. Market opportunities appear or dissolve in months or weeks instead of years.”

Diego Lo Giudice and Dave West, Forrester
February 2011
Transforming Application Delivery

Cha Cha Cha Changes

Agile at Scale with Aligned Autonomy

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



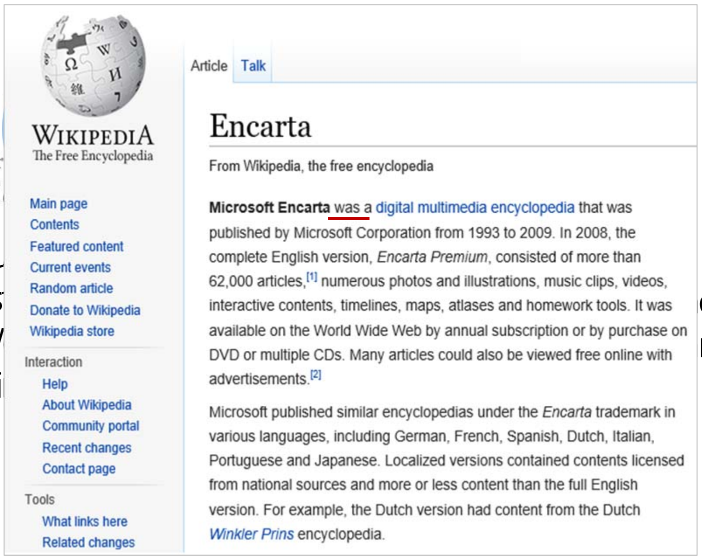
Organization
Roles
Teams
Cadence
Taxonomy
} Alignment

Plan
Practices
} Autonomy

Microsoft

Microsoft
Enc

- Bu
- Es e
- W nt to
- Hi

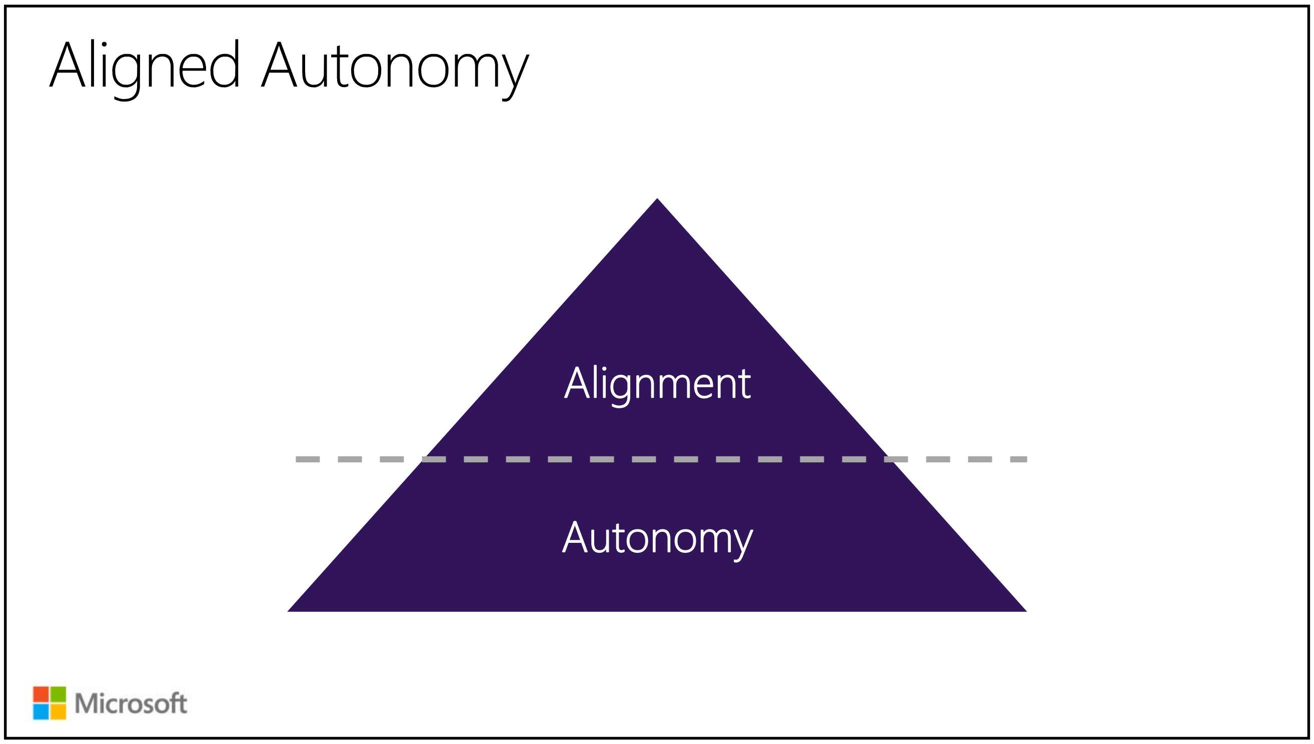


Encarta

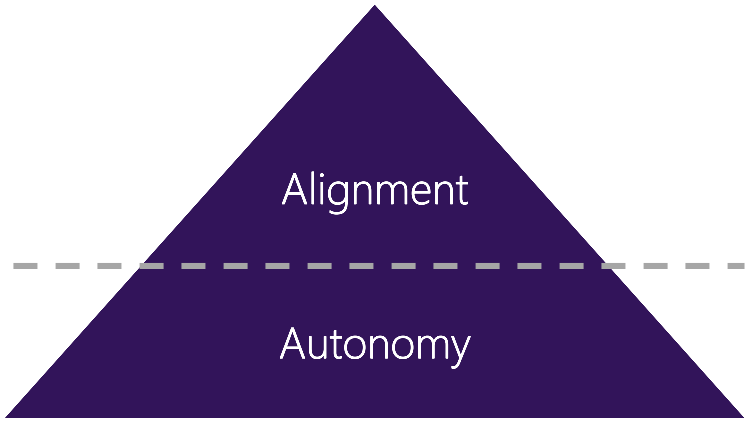
From Wikipedia, the free encyclopedia

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]

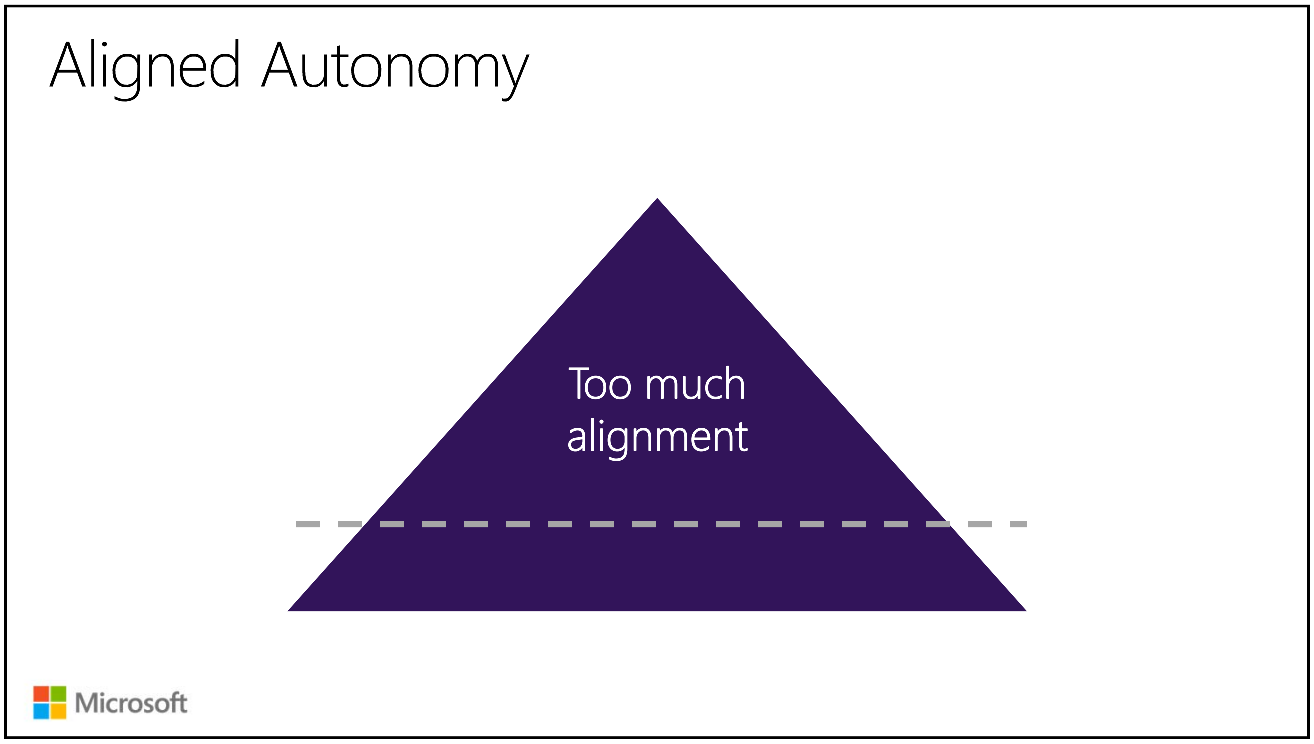
Microsoft published similar encyclopedias under the *Encarta* trademark in various languages, including German, French, Spanish, Dutch, Italian, Portuguese and Japanese. Localized versions contained contents licensed from national sources and more or less content than the full English version. For example,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



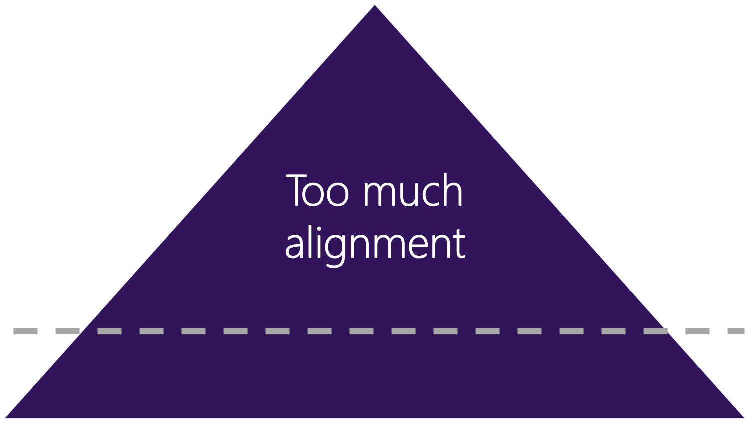
Aligned Autonomy



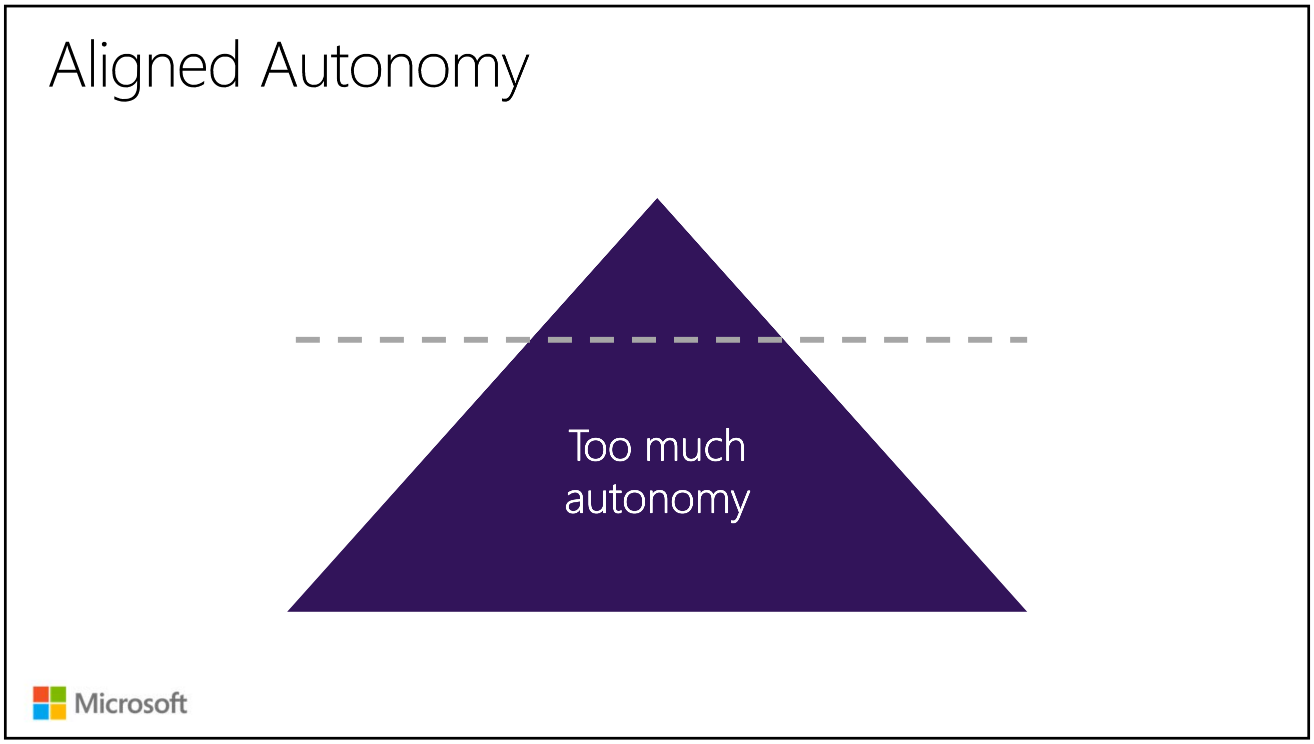
Microsoft



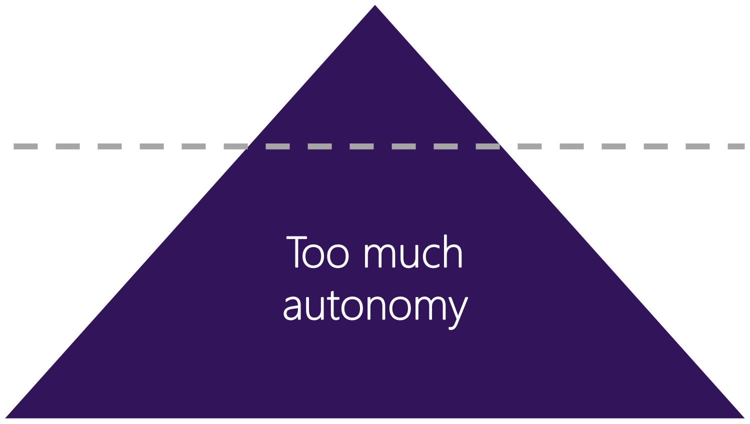
Aligned Autonomy



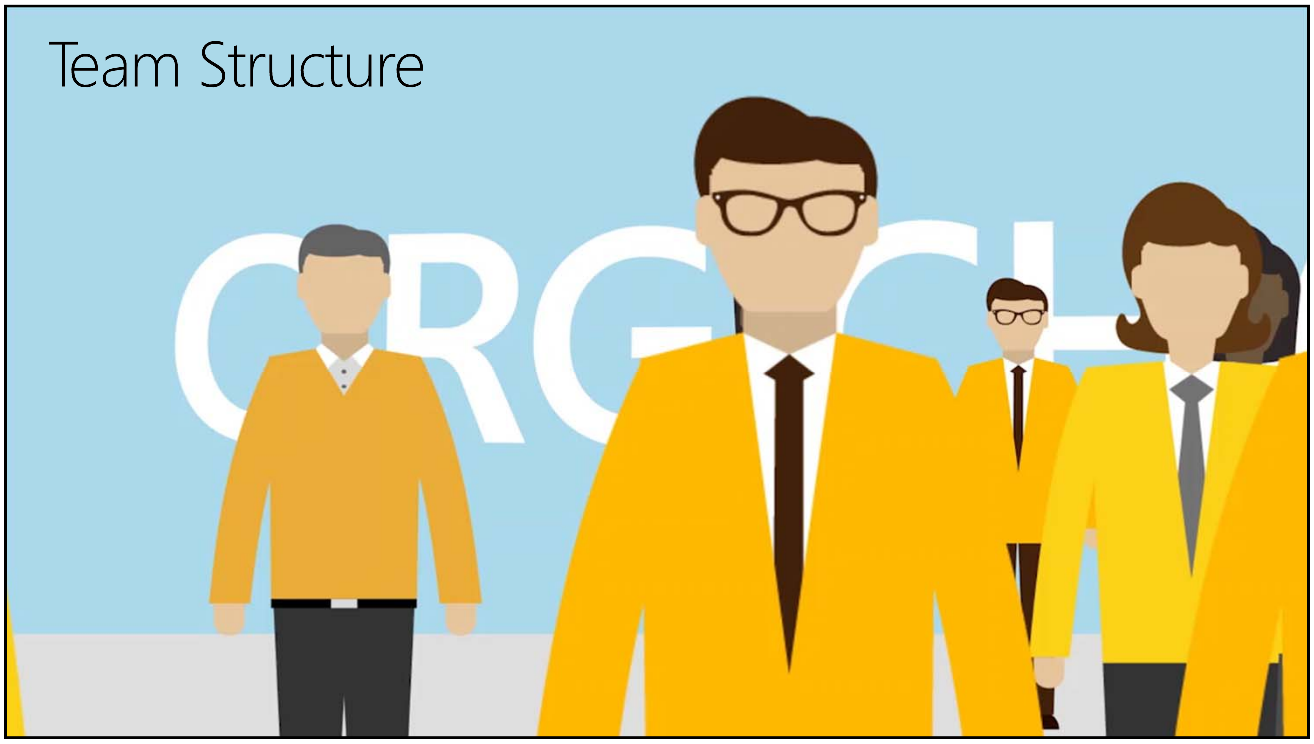
Microsoft



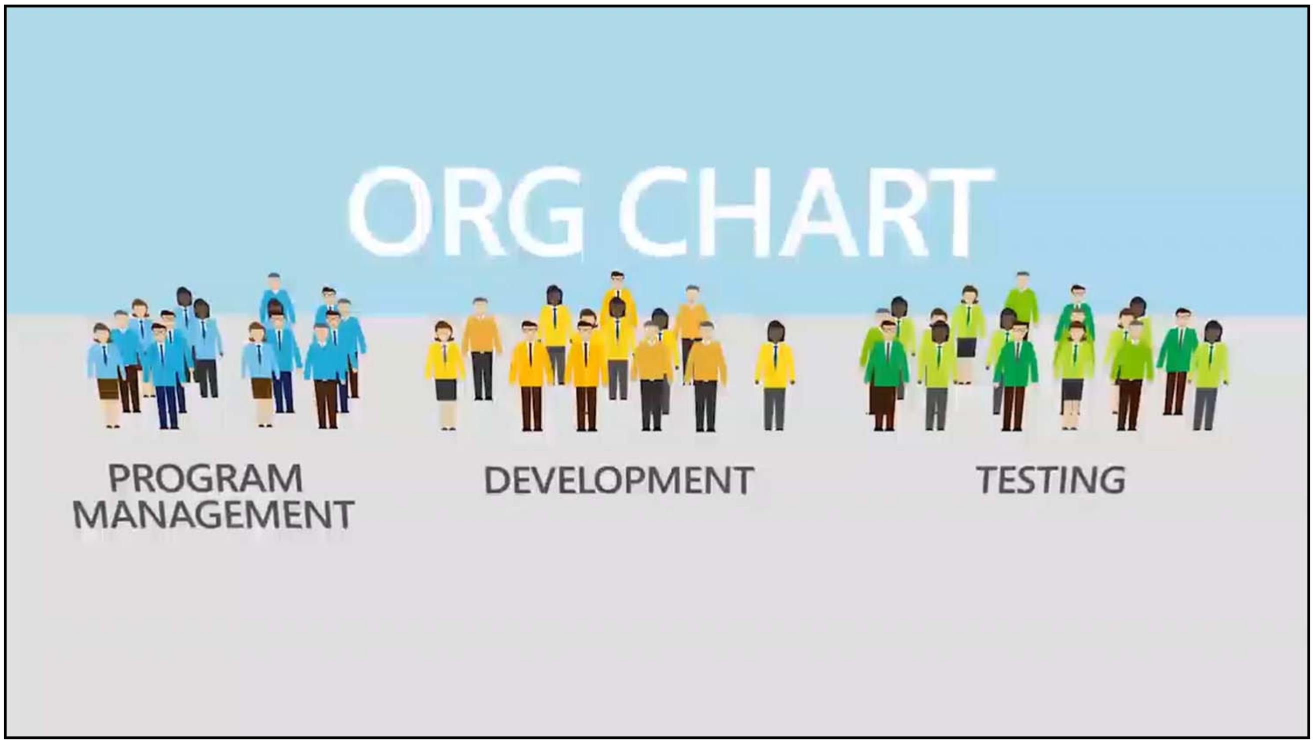
Aligned Autonomy

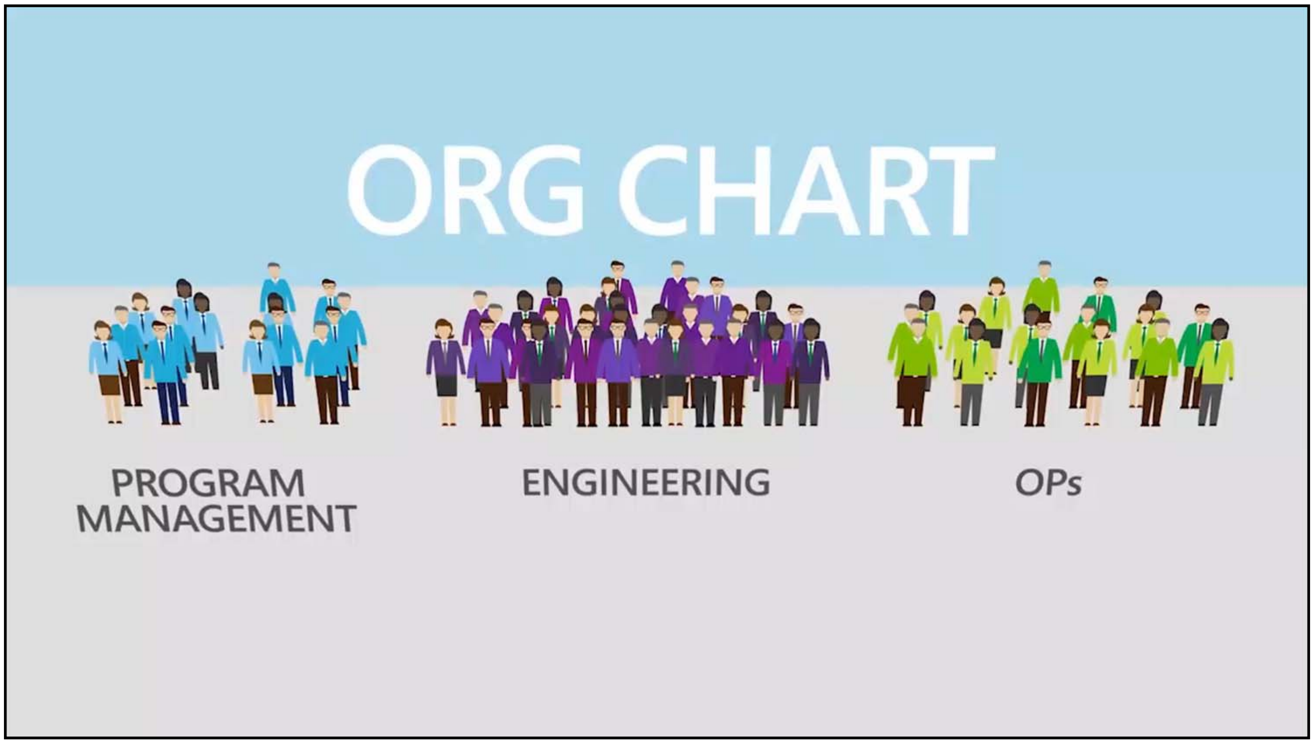


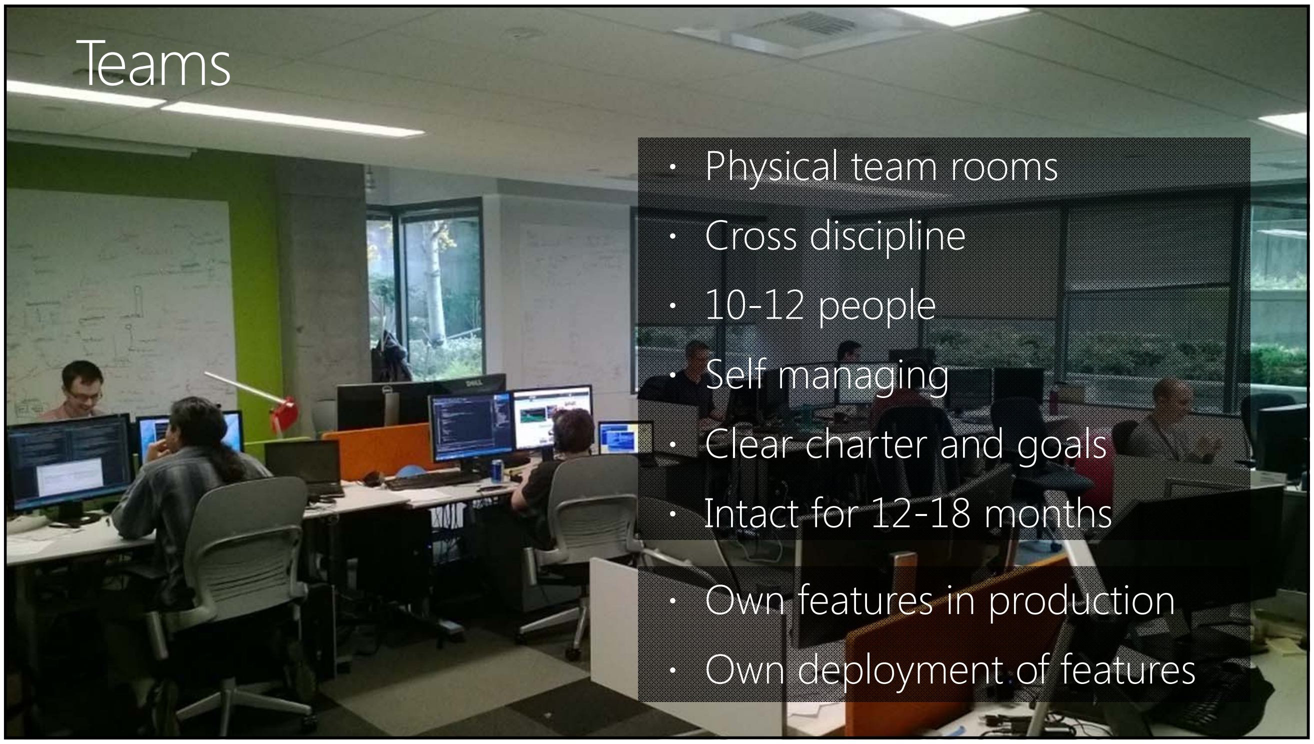
Microsoft



Team Structure


ORG CHART
PROGRAM MANAGEMENT
DEVELOPMENT
TESTING

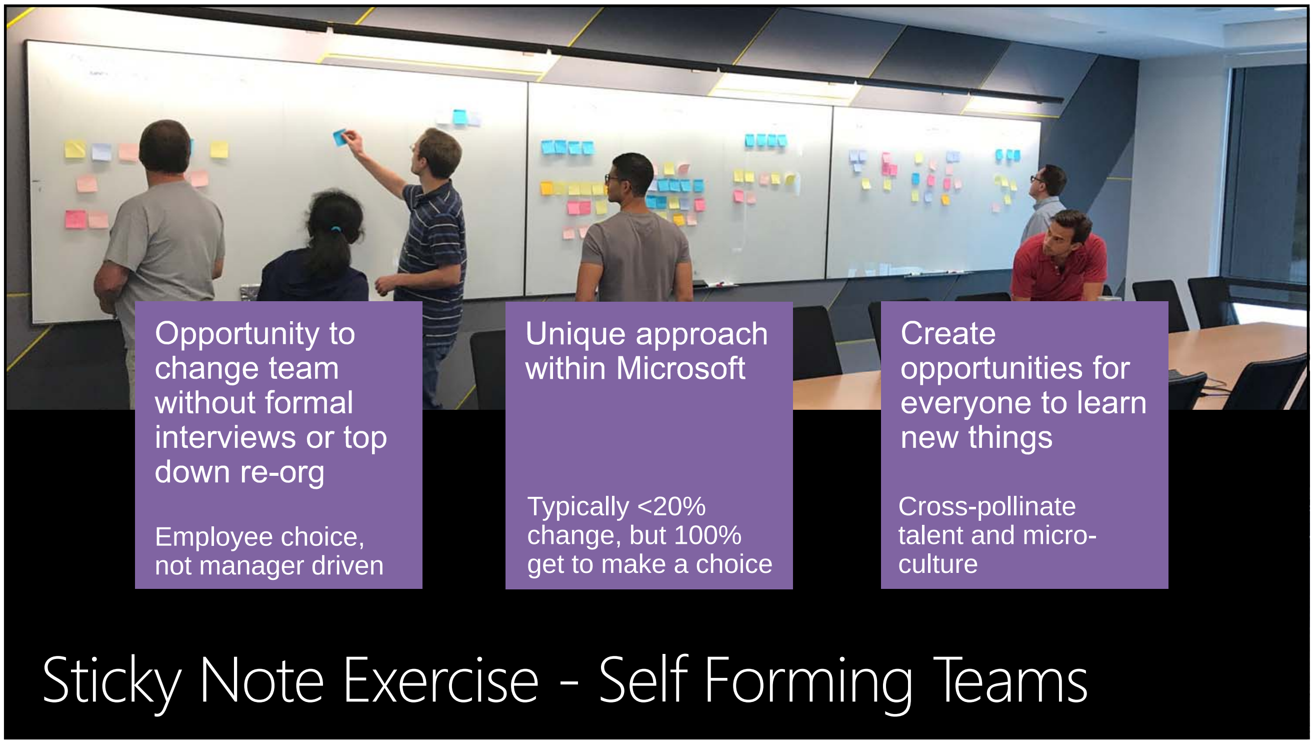

ORG CHART
PROGRAM MANAGEMENT
ENGINEERING
OPs



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

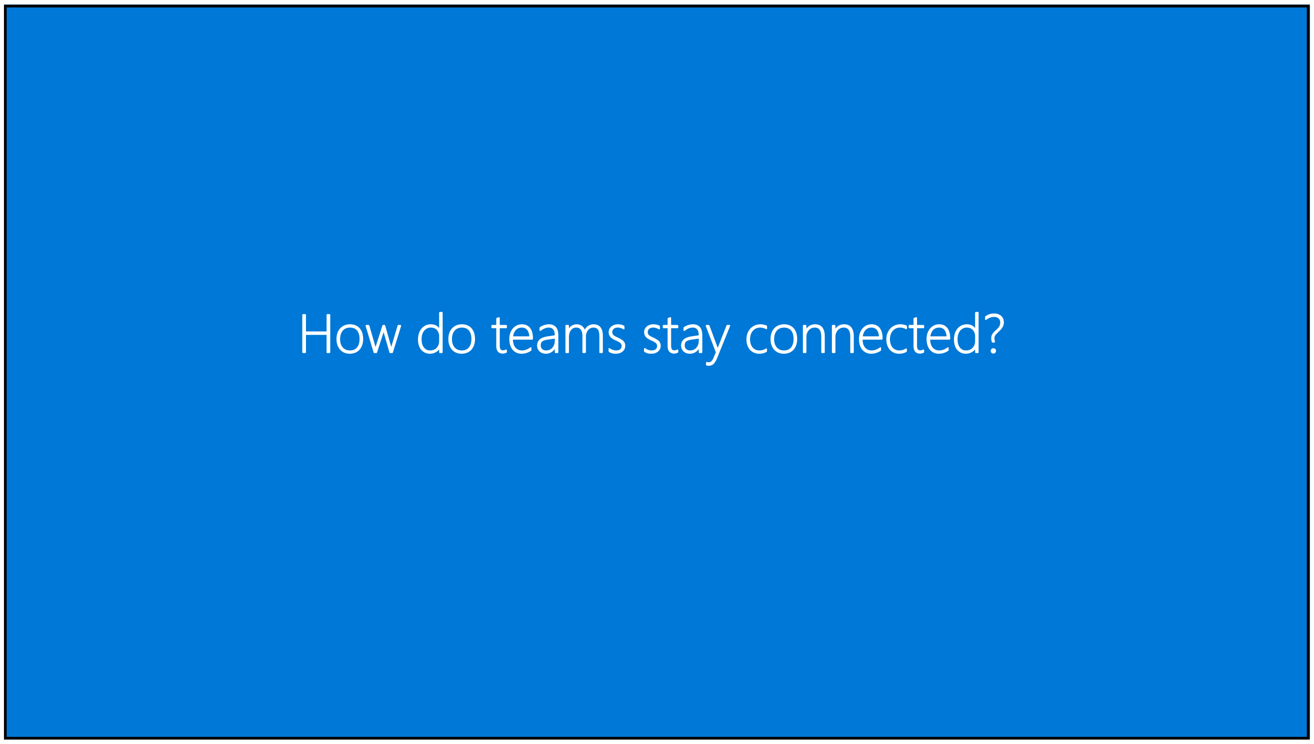
Unique approach within Microsoft

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

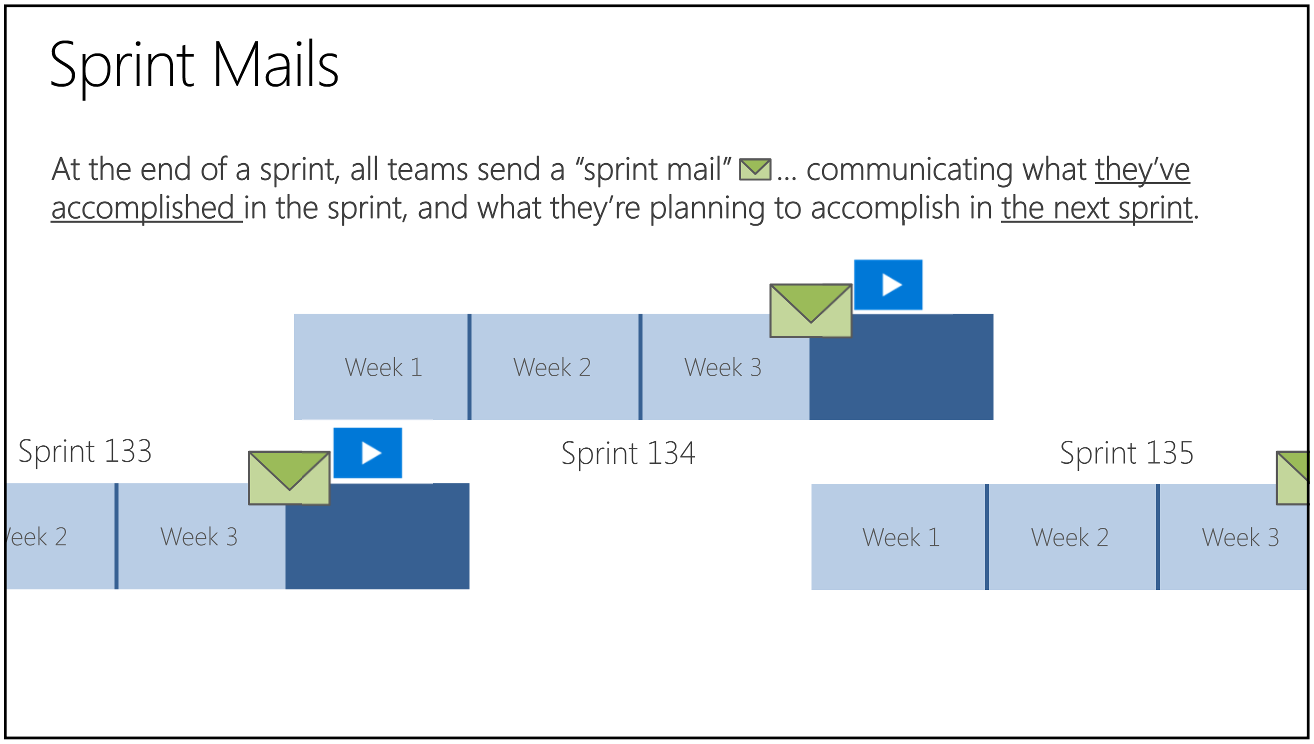
Create opportunities for everyone to learn new things

Cross-pollinate talent and micro-culture

Sticky Note Exercise - Self Forming Teams

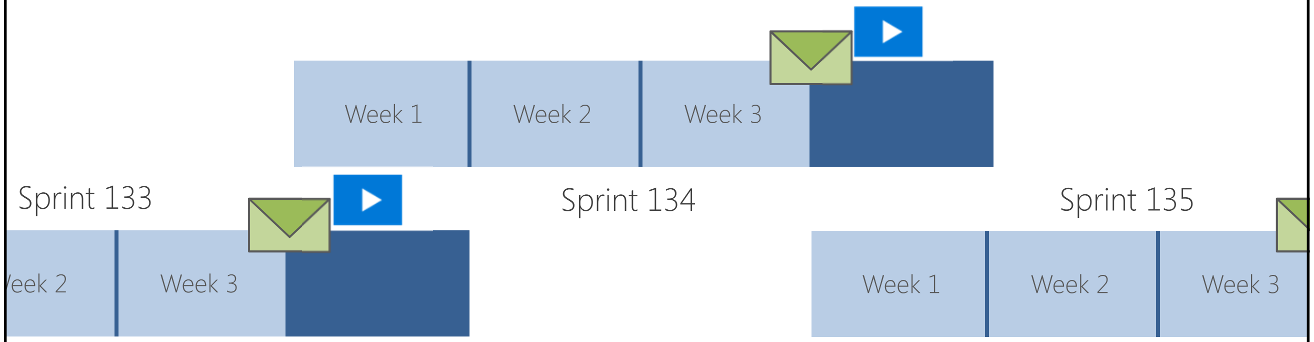


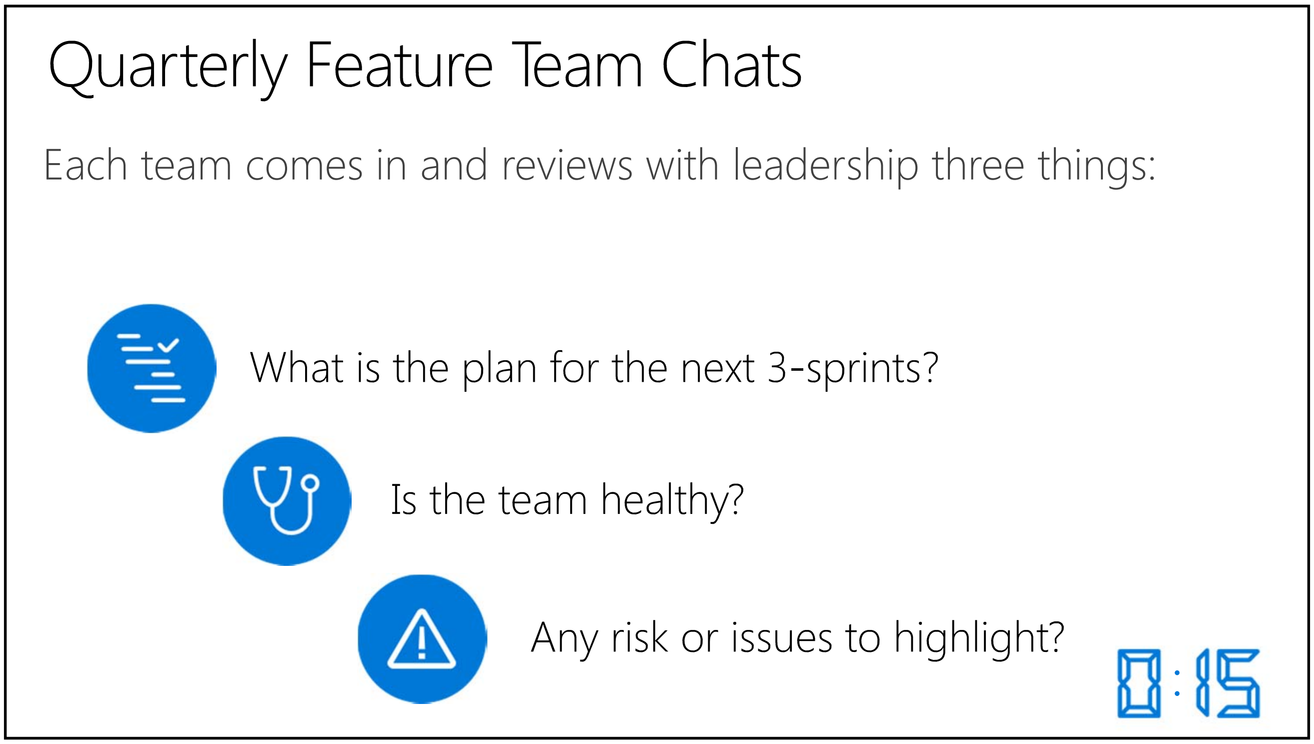
How do teams stay connected?



Sprint Mails

At the end of a sprint, all teams send a "sprint mail" ✉... communicating what they've accomplished in the sprint, and what they're planning to accomplish in the next sprint.



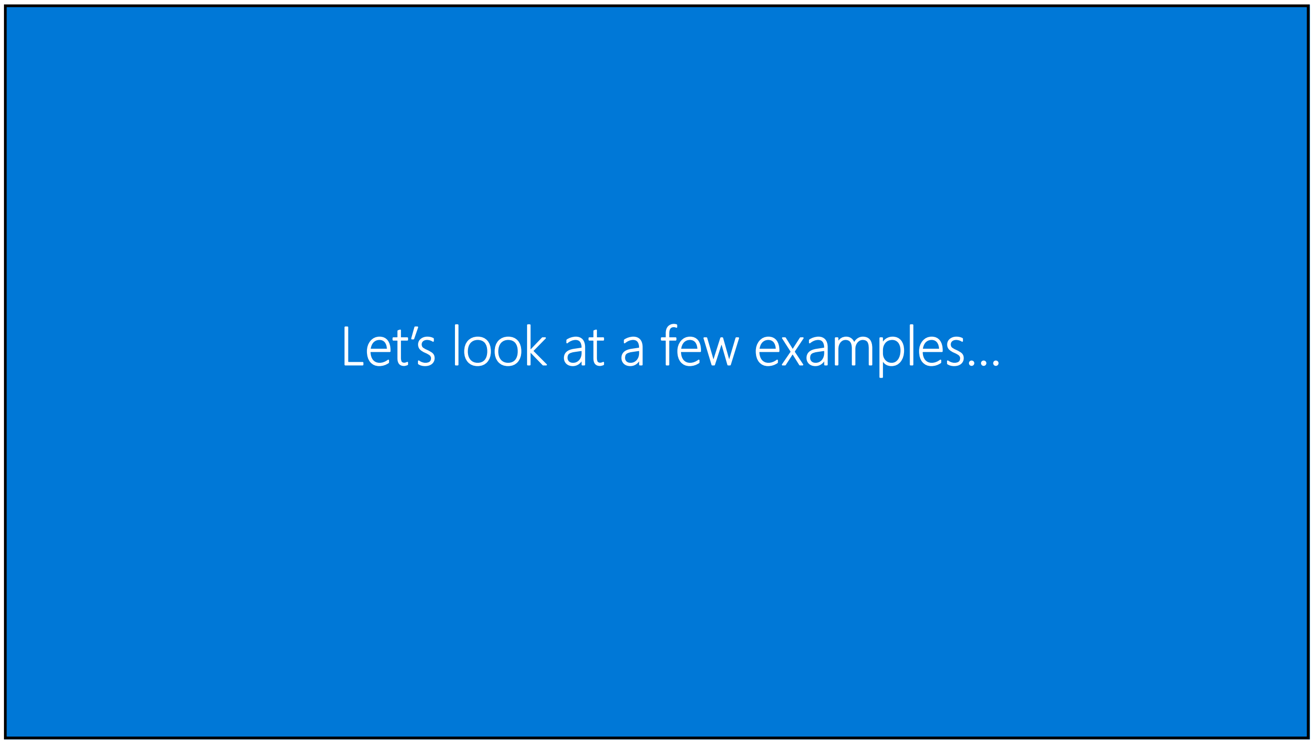


Quarterly Feature Team Chats

Each team comes in and reviews with leadership three things:

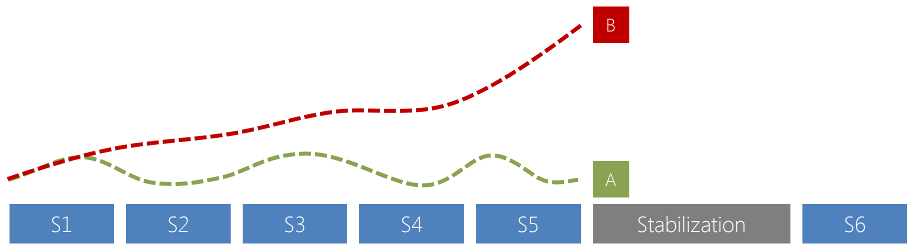
- What is the plan for the next 3-sprints?
- Is the team healthy?
- Any risk or issues to highlight?

0:15



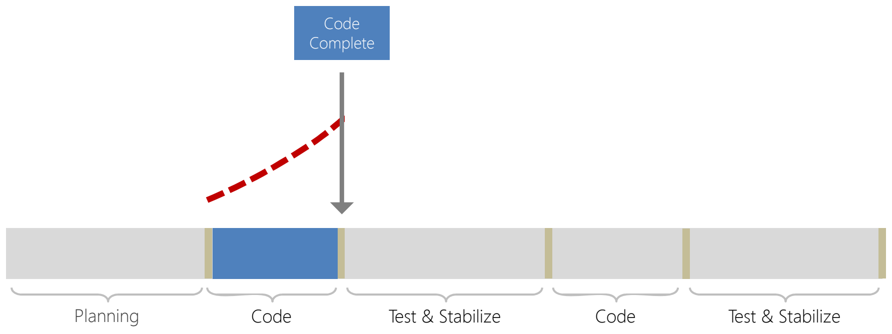
Let’s look at a few examples…

The “Stabilization” Sprint

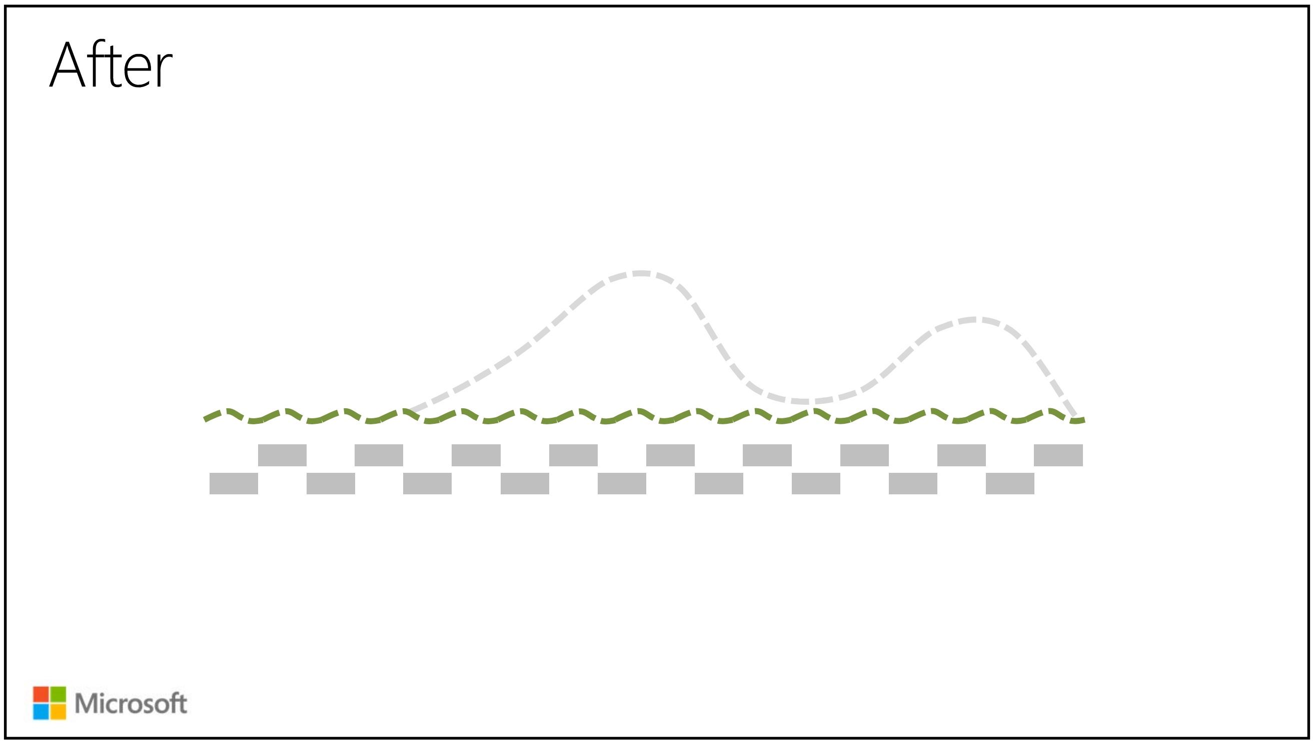


“Let’s do this Agile thing… but we should probably reserve some time to stabilize things.”

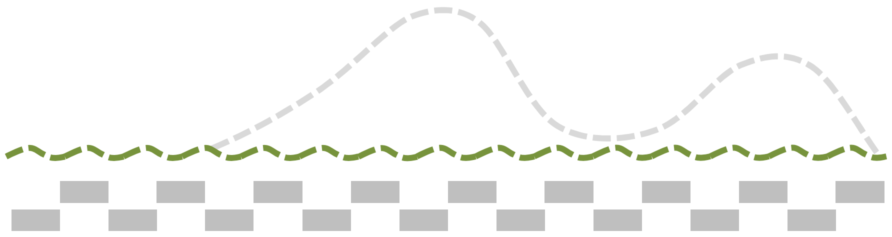
Microsoft



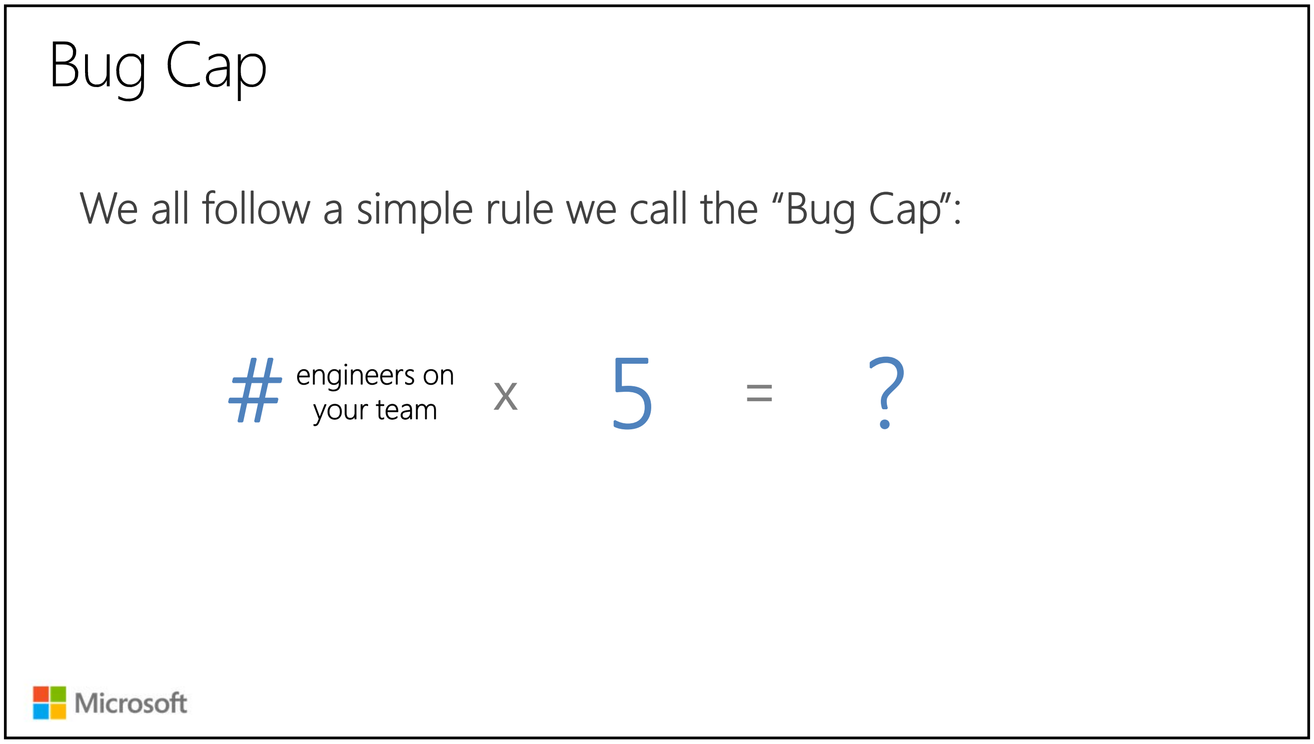
Microsoft



After



Microsoft



Bug Cap

We all follow a simple rule we call the "Bug Cap":

# engineers on your team x 5 = ?

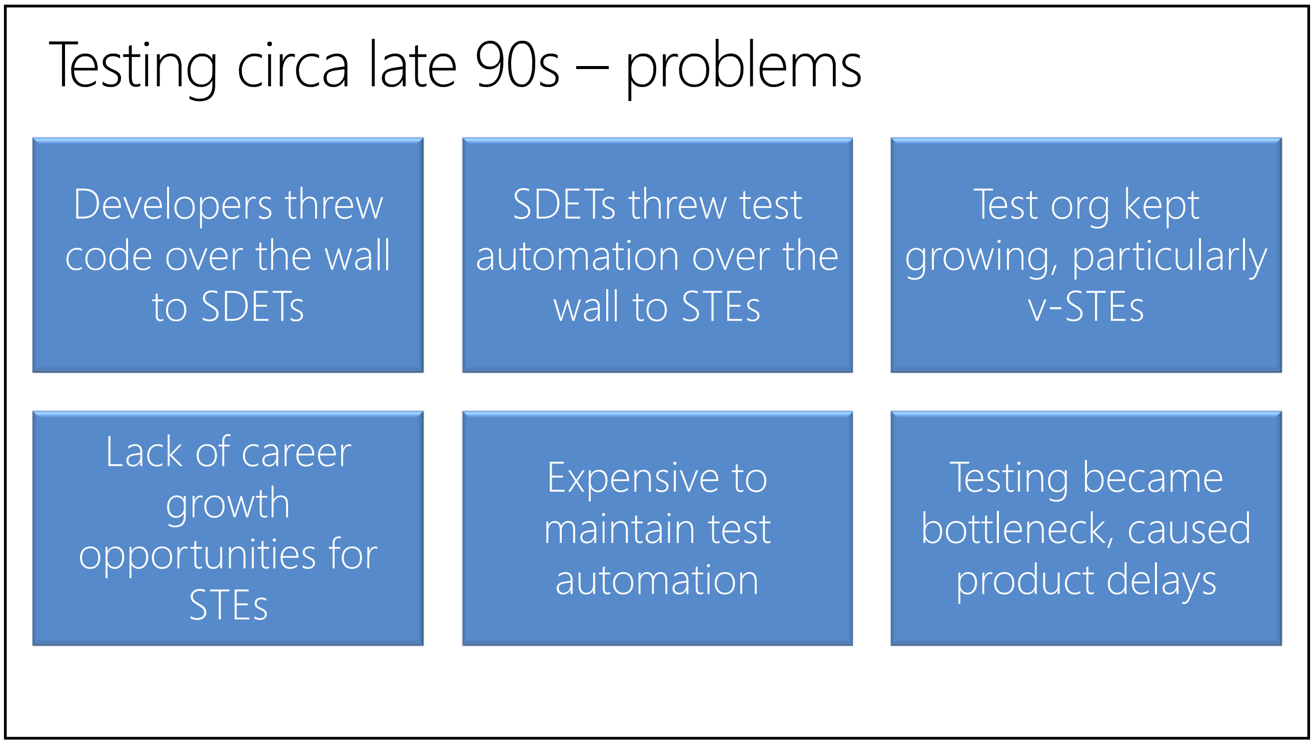
Microsoft

Shift-left on Quality

Testing at Microsoft circa 1990s

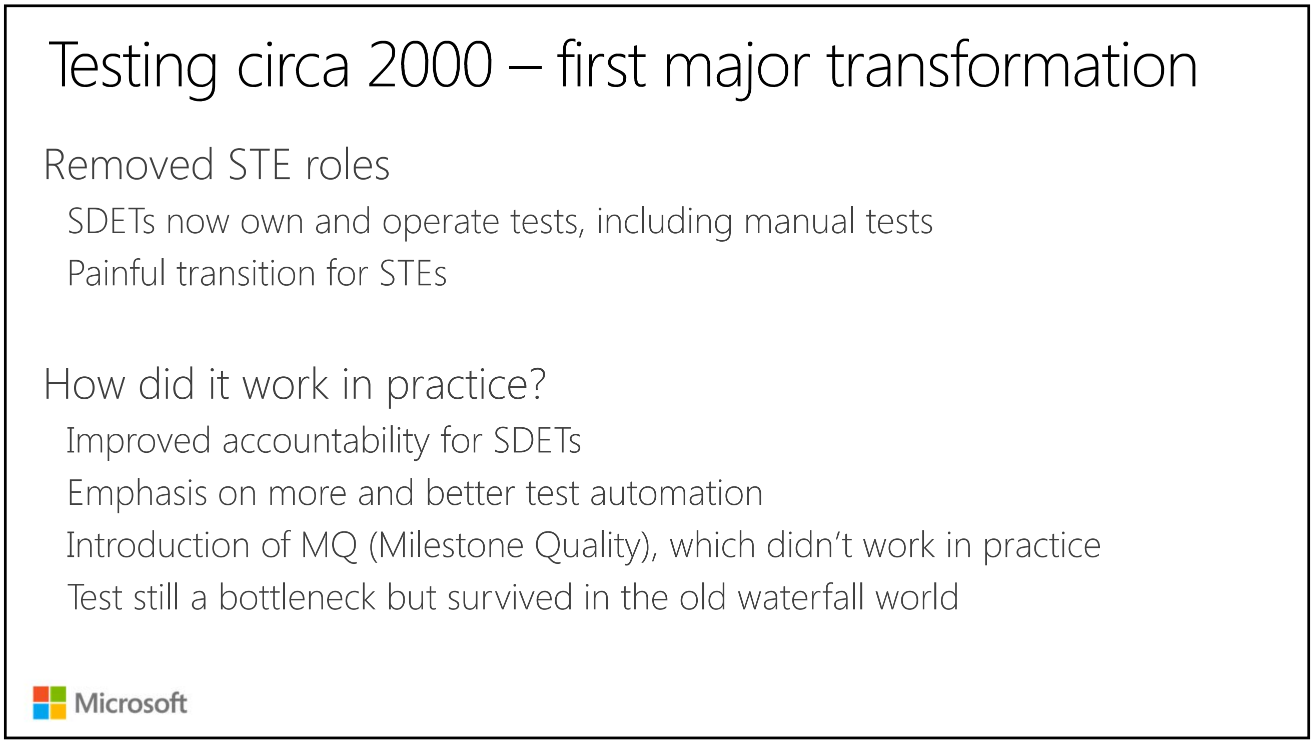
- Three distinct disciplines in every product team
  - Developer, Tester, PM (Ratio - 1 : 1 : 0.5)
- Test had two roles
  - Software Design Engineer in Test (SDET) – developed automation & test infra
  - Software Test Engineer (STE) – ran automated and manual tests
- How did it work in practice?
  - Worked reasonably well in the beginning; Microsoft achieved commercial success with Windows and Office
  - Product signoff based on formal Quality measurements
  - Developed deep expertise Testing techniques and tooling

Microsoft



Testing circa late 90s – problems

- Developers threw code over the wall to SDETs
- SDETs threw test automation over the wall to STEs
- Test org kept growing, particularly v-STEs
- Lack of career growth opportunities for STEs
- Expensive to maintain test automation
- Testing became bottleneck, caused product delays



Testing circa 2000 – first major transformation

Removed STE roles

- SDETs now own and operate tests, including manual tests
- Painful transition for STEs

How did it work in practice?

- Improved accountability for SDETs
- Emphasis on more and better test automation
- Introduction of MQ (Milestone Quality), which didn't work in practice
- Test still a bottleneck but survived in the old waterfall world

Microsoft

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

Testing in Cloud cadence – problems

New problems emerged, old ones exacerbated

- Testing became major bottleneck – we reached a breaking point
- Trains didn't run on time
- Lack of accountability on the Developers – no real incentive to change
- High frustration among SDETs. Major retention issues.
- ….

Our model was broken

- Bing, being first major cloud service at Microsoft, noticed it first
- Over next few years, every team at Microsoft moved to the Cloud and changed their testing approach

Our problems: September 2014

Tests took too long

Over 22 hours for nightly run

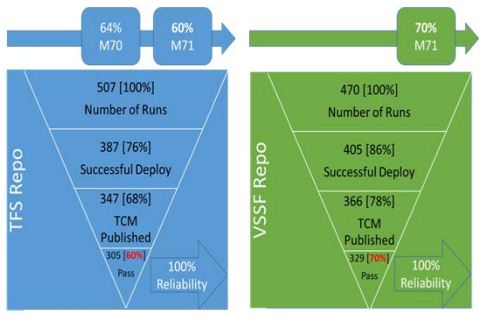
2 days for the full run

Tests failed frequently

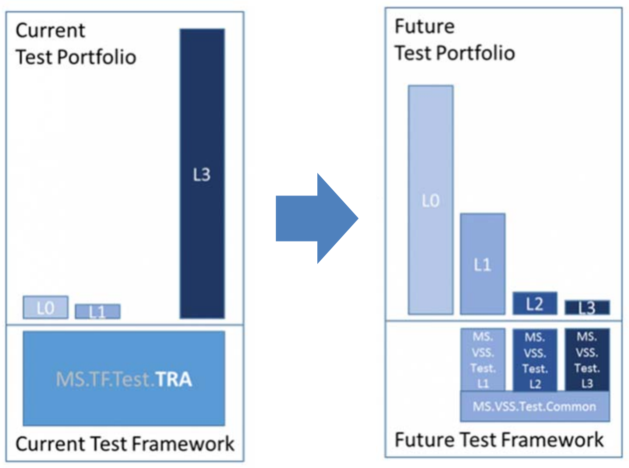
Only ~60% of P0 runs passed 100%; Each NAR suite had many failures

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

We introduced a finer-grained test classification scheme

Levels can roughly be understood as a measure of external dependencies

L0/L1 - Unit tests

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

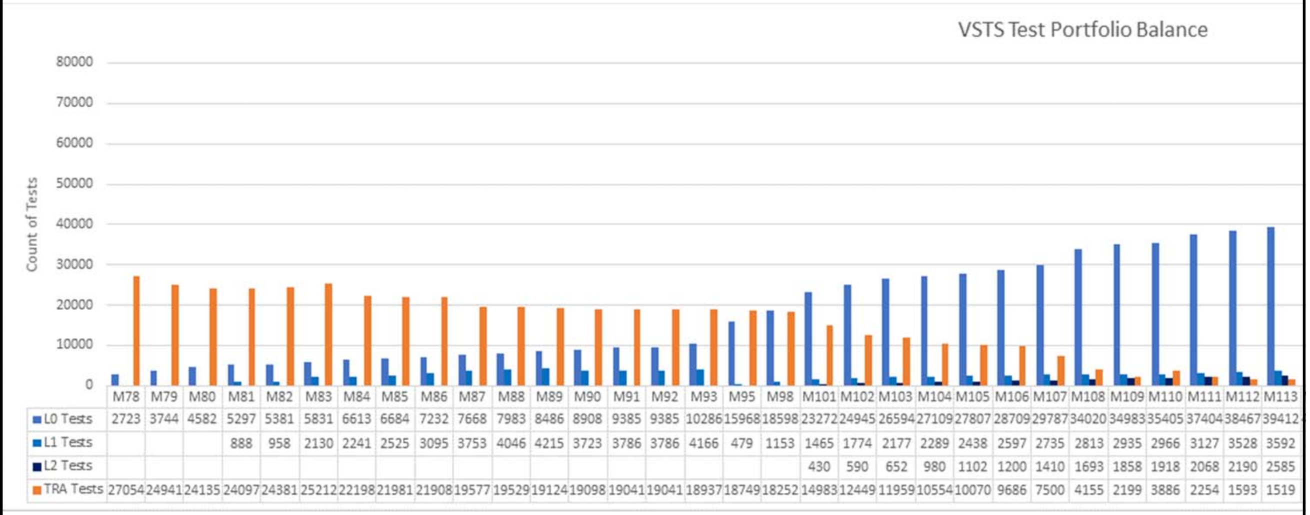
L2/L3 - Functional tests

- L2 – Functional tests run against "testable" service deployment
- L3 – Restricted class integration tests that run against production

SelfTest Suite – L2/L3 functional Tests (Priority 0)

SelfHost Suite – L2/L3 functional Tests (Priority > 0)

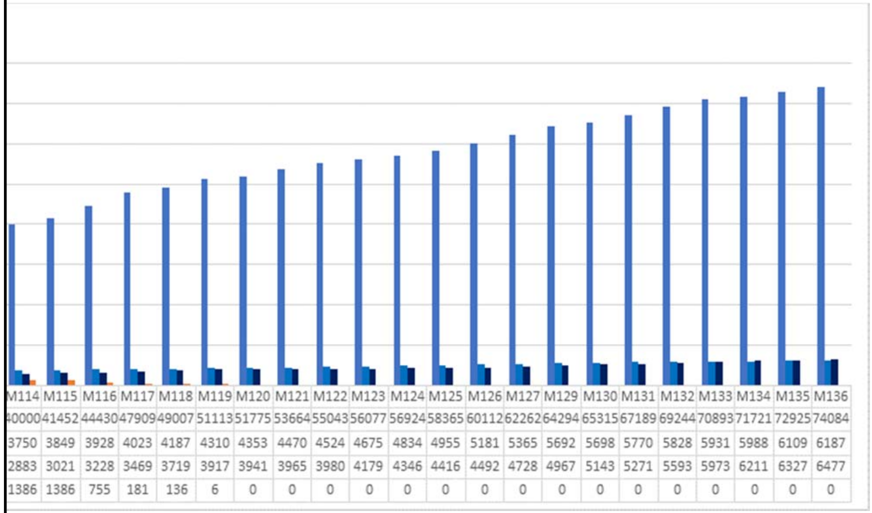
Test portfolio over time



	M78	M79	M80	M81	M82	M83	M84	M85	M86	M87	M88	M89	M90	M91	M92
■ L0 Tests	2723	3744	4582	5297	5381	5831	6613	6684	7232	7668	7983	8486	8908	9385	9385
■ L1 Tests				888	958	2130	2241	2525	3095	3753	4046	4215	3723	3786	3786
■ L2 Tests															
■ TRA Tests	27054	24941	24135	24097	24381	25212	22198	21981	21908	19577	19529	19124	19098	19041	19041

	M93	M95	M98	M101	M102	M103	M104	M105	M106	M107	M108	M109	M110	M111	M112	M113
■ L0 Tests	10286	15968	18598	23272	24945	26594	27109	27807	28709	29787	34020	34983	35405	37404	38467	39412
■ L1 Tests	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	18937	18749	18252	14983	12449	11959	10554	10070	9686	7500	4155	2199	3886	2254	1593	1519

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

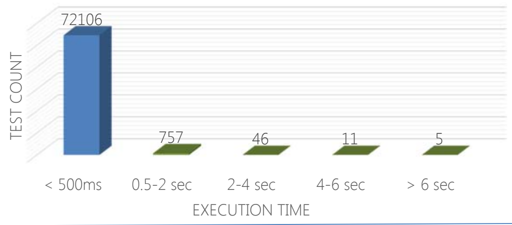
TYPE	M78	M136	DELTA
L0	2723	74084	+ 71,361
L1		6187	+ 6,187
L2		6477	+ 6,477
TRA	27054	0	- 27,054

Continuous focus on test execution time

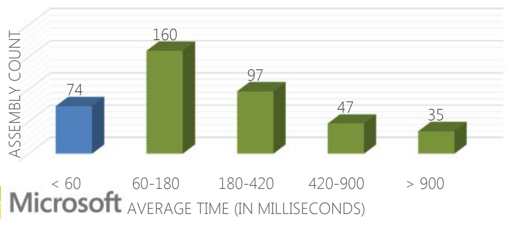
Level 0

- Execution: 6:00 mins
- Total Tests: ~72925
- Number of Assemblies: 2104

L0 tests by execution time



L0 Assemblies' Average Time chart

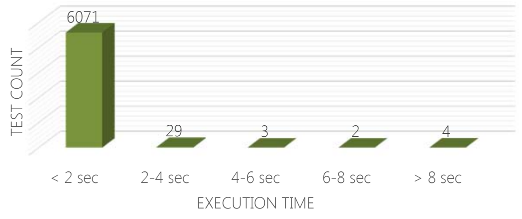


Microsoft

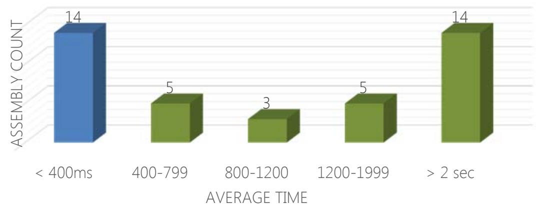
Level 1

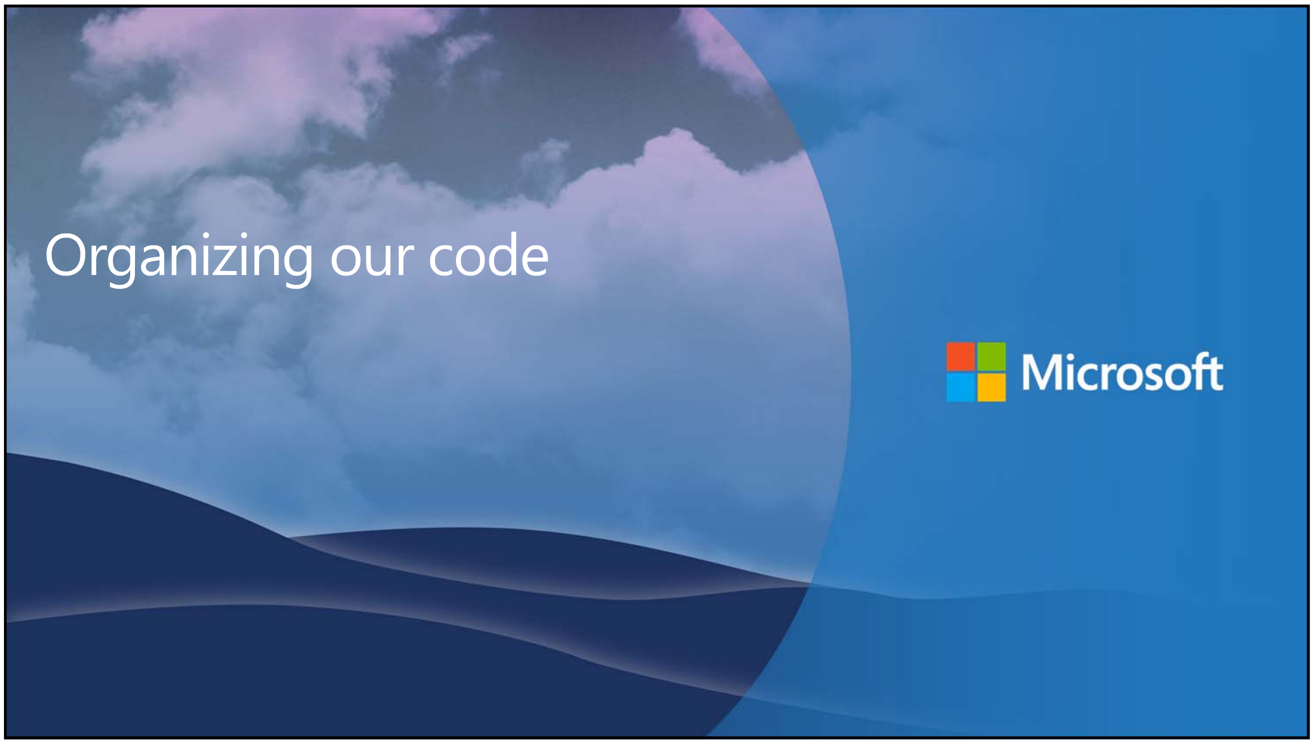
- Execution: 3:30 mins
- Total Tests: ~6109
- Number of Assemblies: 41

L1 tests by execution time



L1 Assemblies' Average Time chart



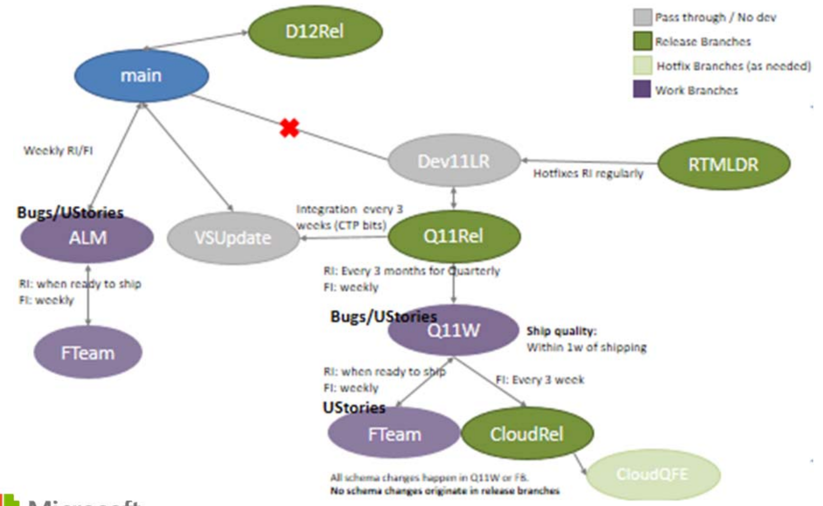


Organizing our code

Microsoft

Traditional branch structure

The OLD way



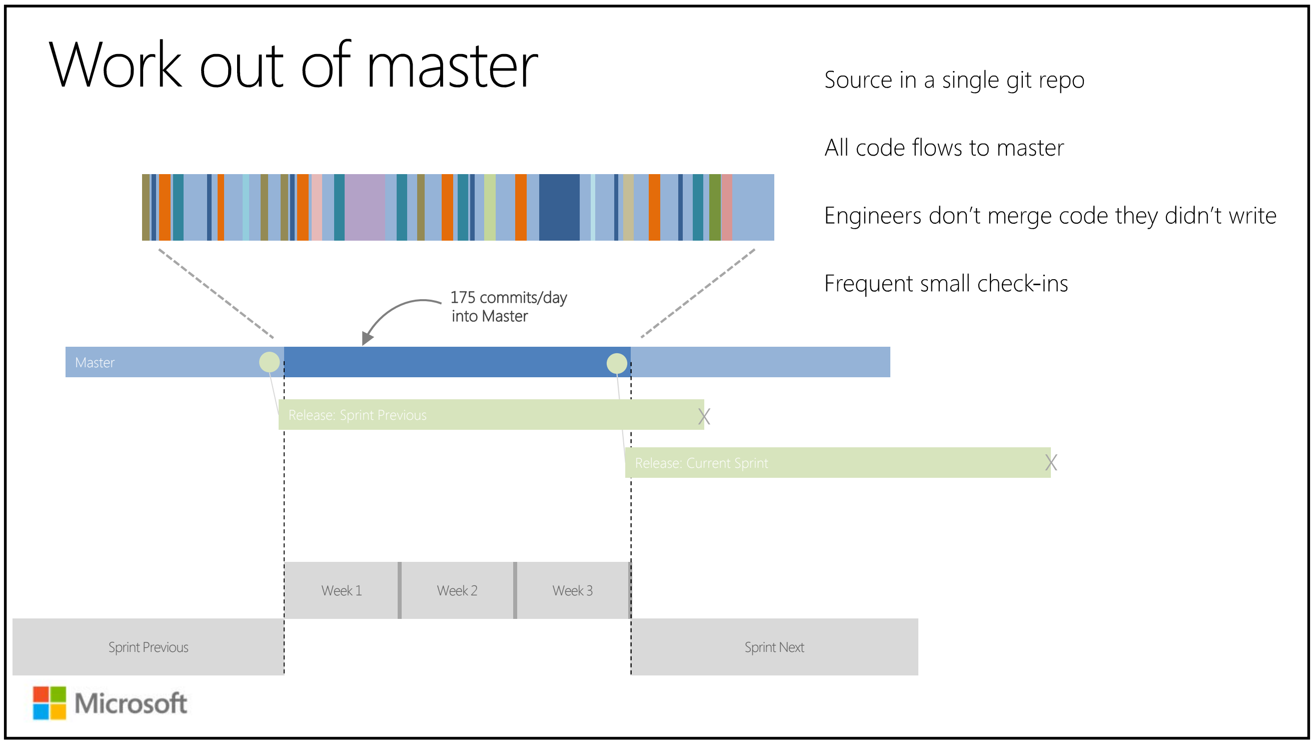
Deep branch hierarchy

Creates merge and integration debt

Significant costs to code flow

Complex logistics Engineers must understand

Microsoft



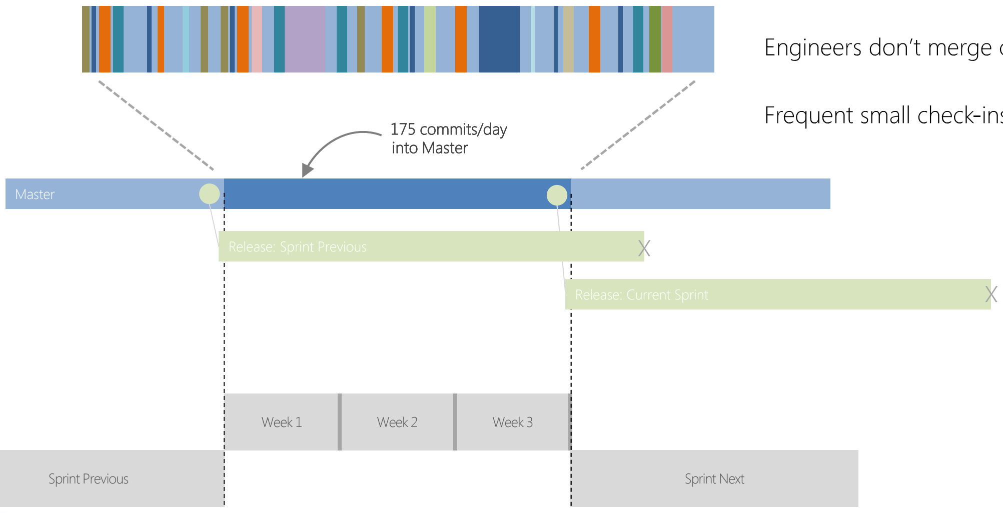
Work out of master

Source in a single git repo

All code flows to master

Engineers don't merge code they didn't write

Frequent small check-ins



Microsoft

Feature Flags

Controlling exposure in the cloud

Microsoft

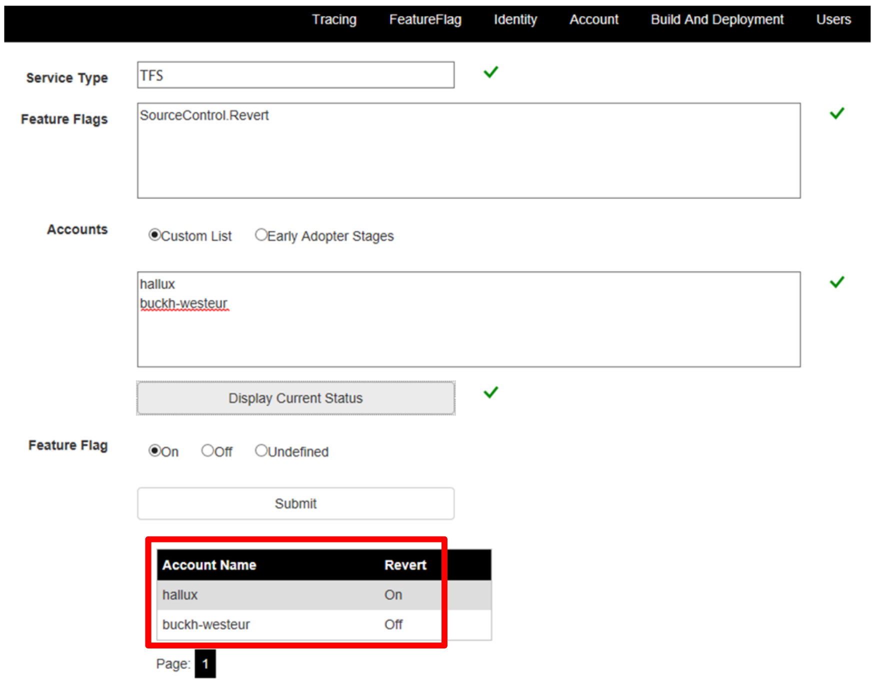
What do feature flags give us?

- 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

Microsoft

Control

- PowerShell
  - Get-FeatureFlag
  - Set-FeatureFlag
- Web UI



Service Type: TFS ✓

Feature Flags: SourceControl.Revert ✓

Accounts: [✓ Custom List] Early Adopter Stages

hallux
buckh-westeur ✓

Display Current Status ✓

Feature Flag: [✓ On] Off Undefined

Submit

Account Name	Revert
hallux	On
buckh-westeur	Off

Page: 1

Microsoft

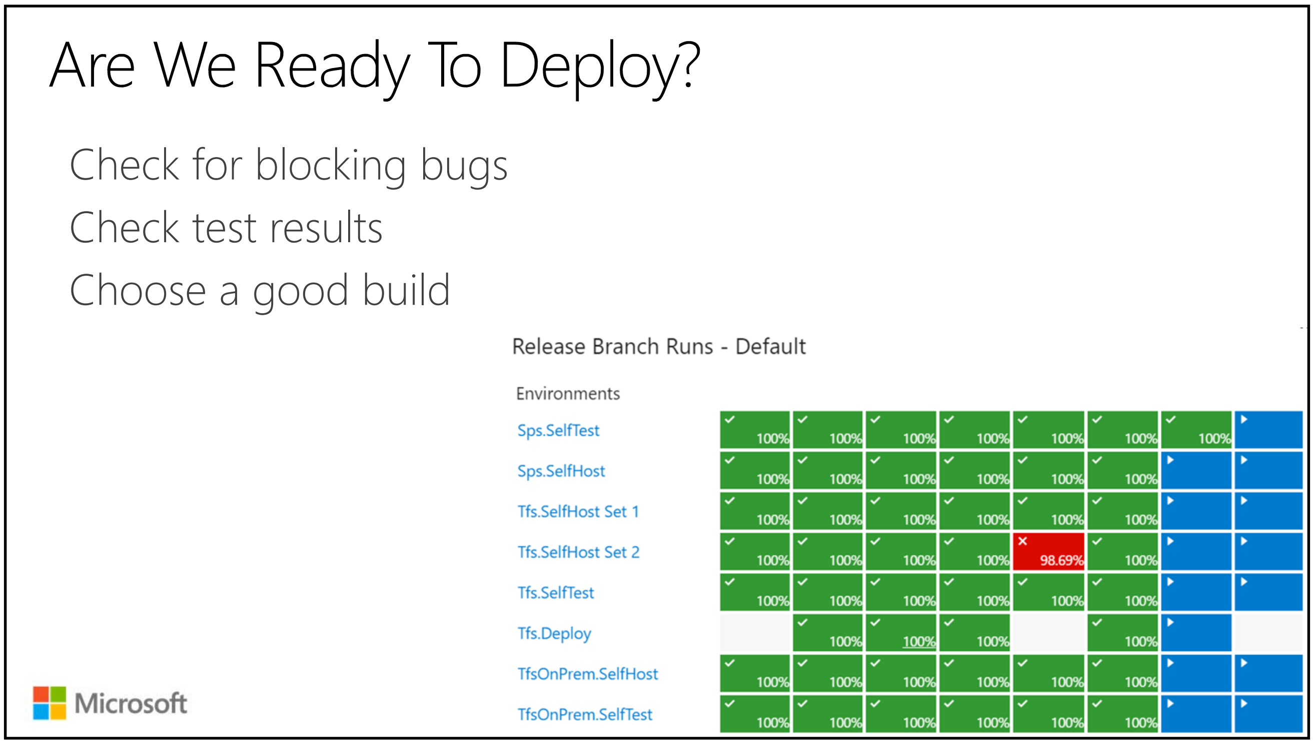
Deployment Practices

Microsoft

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

Microsoft



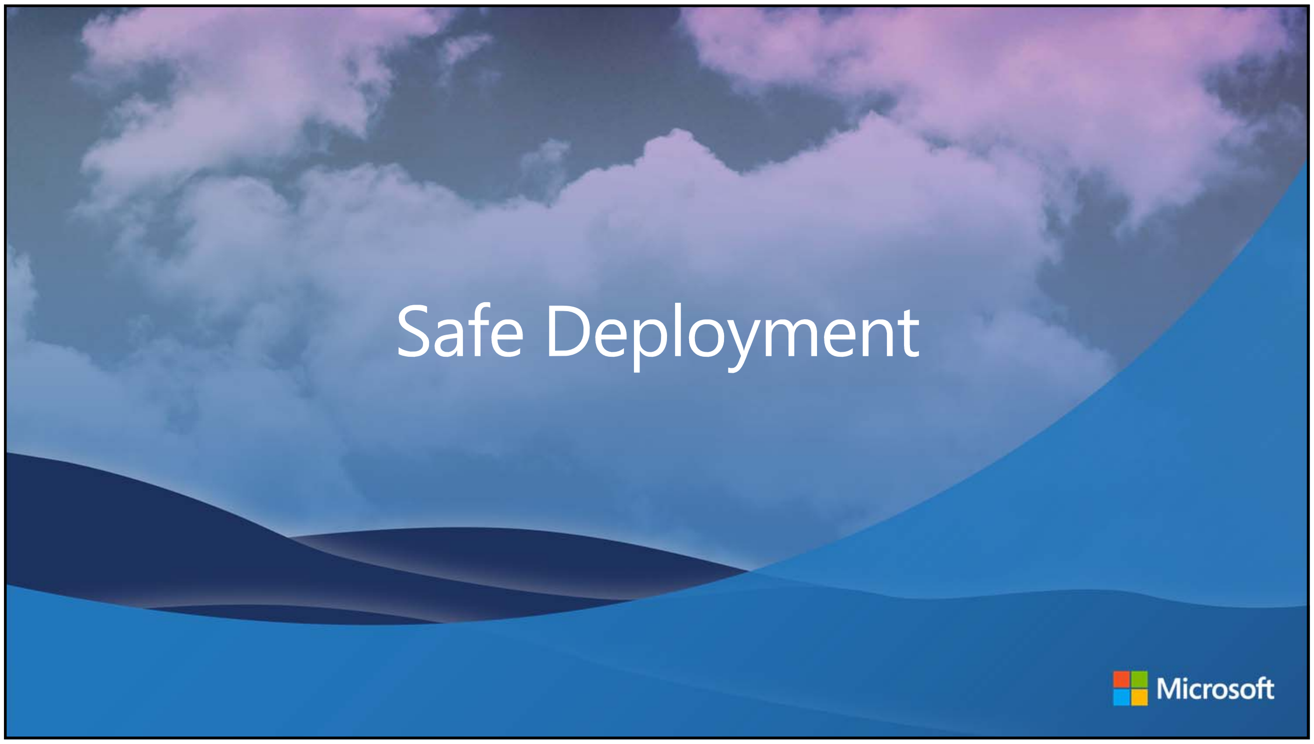
Are We Ready To Deploy?

Check for blocking bugs
Check test results
Choose a good build

Release Branch Runs - Default

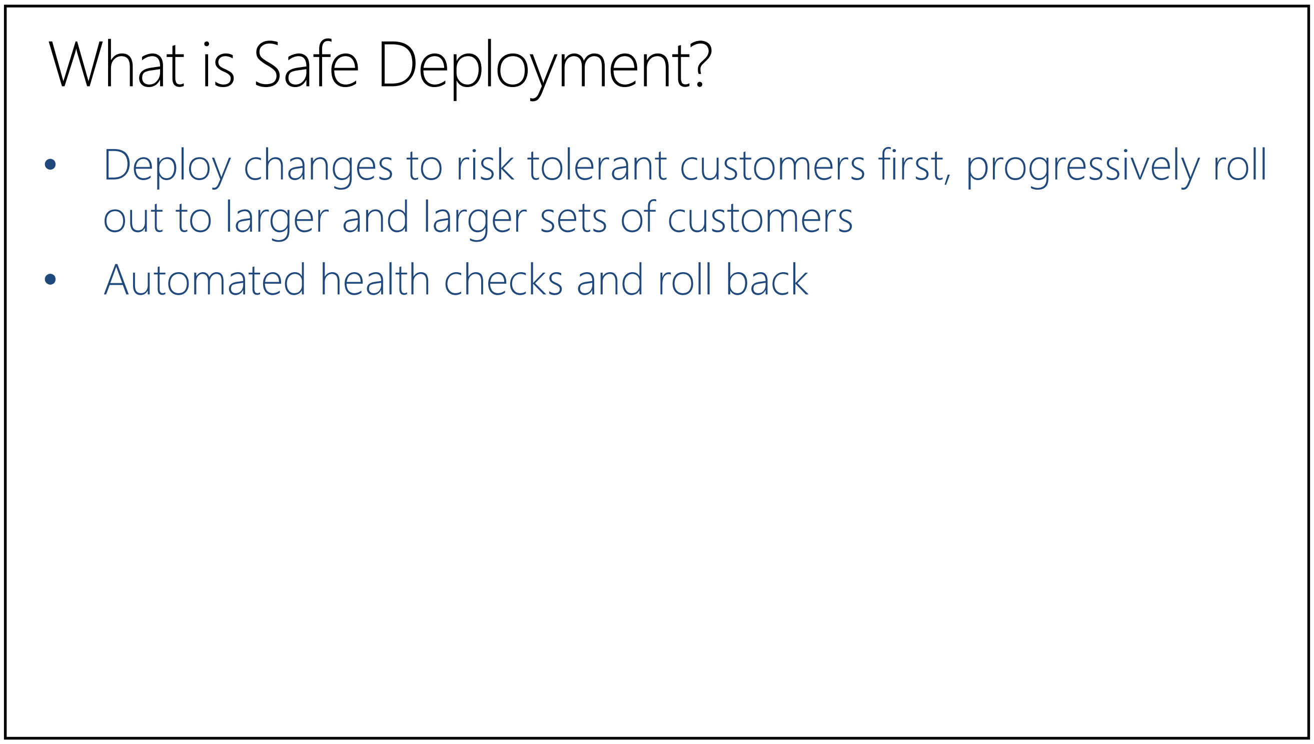
Environments								
Sps.SelfTest	✓ 100%	✓ 100%	✓ 100%	✓ 100%	✓ 100%	✓ 100%	✓ 100%	▶
Sps.SelfHost	✓ 100%	✓ 100%	✓ 100%	✓ 100%	✓ 100%	✓ 100%	▶	▶
Tfs.SelfHost Set 1	✓ 100%	✓ 100%	✓ 100%	✓ 100%	✓ 100%	✓ 100%	▶	▶
Tfs.SelfHost Set 2	✓ 100%	✓ 100%	✓ 100%	✓ 100%	✕ 98.69%	✓ 100%	▶	▶
Tfs.SelfTest	✓ 100%	✓ 100%	✓ 100%	✓ 100%	✓ 100%	✓ 100%	▶	▶
Tfs.Deploy		✓ 100%	✓ 100%	✓ 100%		✓ 100%	▶	
TfsOnPrem.SelfHost	✓ 100%	✓ 100%	✓ 100%	✓ 100%	✓ 100%	✓ 100%	▶	▶
TfsOnPrem.SelfTest	✓ 100%	✓ 100%	✓ 100%	✓ 100%	✓ 100%	✓ 100%	▶	▶

Microsoft



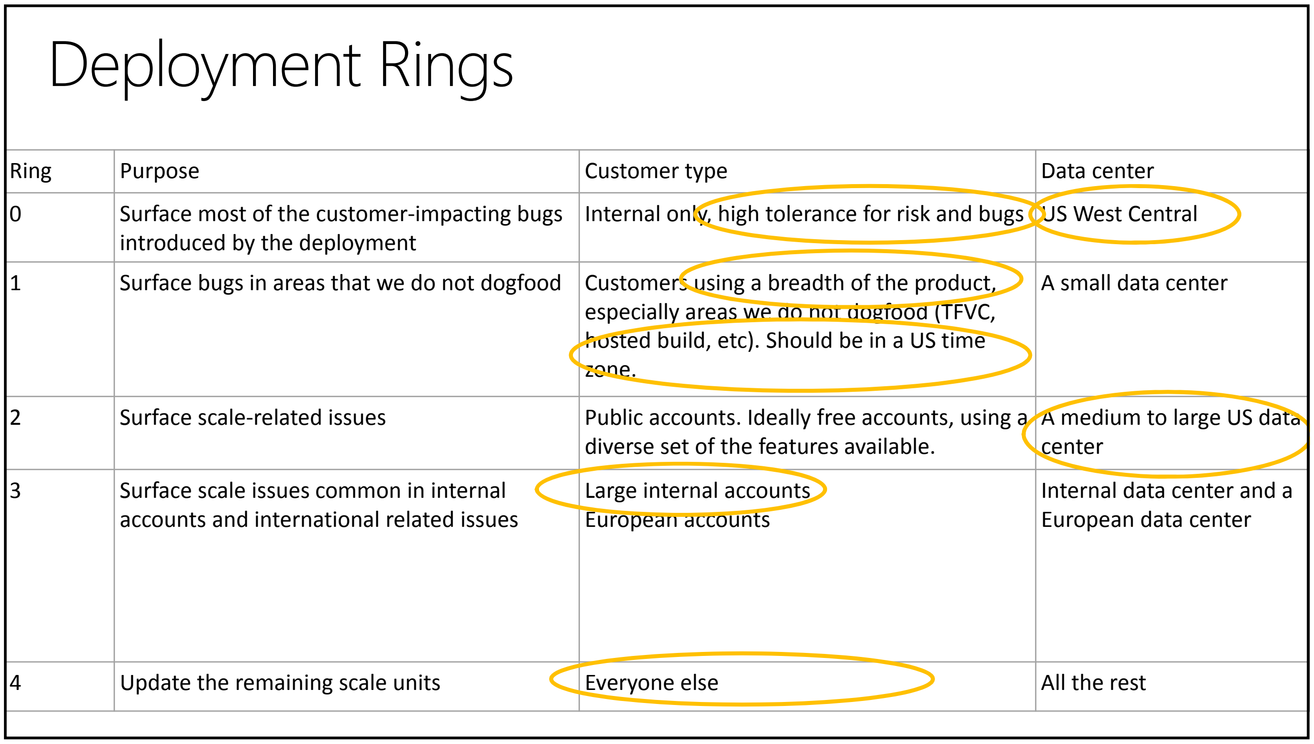
Safe Deployment

Microsoft



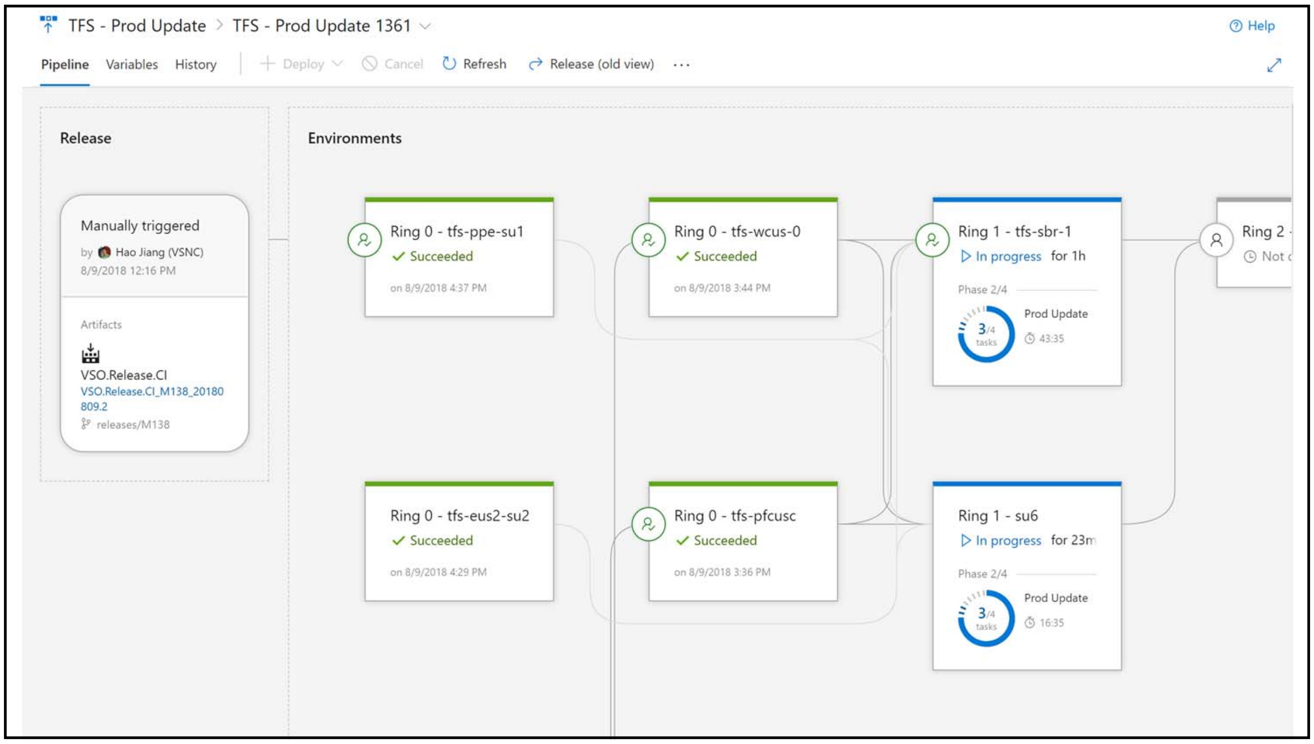
What is Safe Deployment?

- 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

Ring	Purpose	Customer type	Data center
0	Surface most of the customer-impacting bugs introduced by the deployment	Internal only, high tolerance for risk and bugs	US West Central
1	Surface bugs in areas that we do not dogfood	Customers using a breadth of the product, especially areas we do not dogfood (TFVC, hosted build, etc). Should be in a US time zone.	A small data center
2	Surface scale-related issues	Public accounts. Ideally free accounts, using a diverse set of the features available.	A medium to large US data center
3	Surface scale issues common in internal accounts and international related issues	Large internal accounts European accounts	Internal data center and a European data center
4	Update the remaining scale units	Everyone else	All the rest



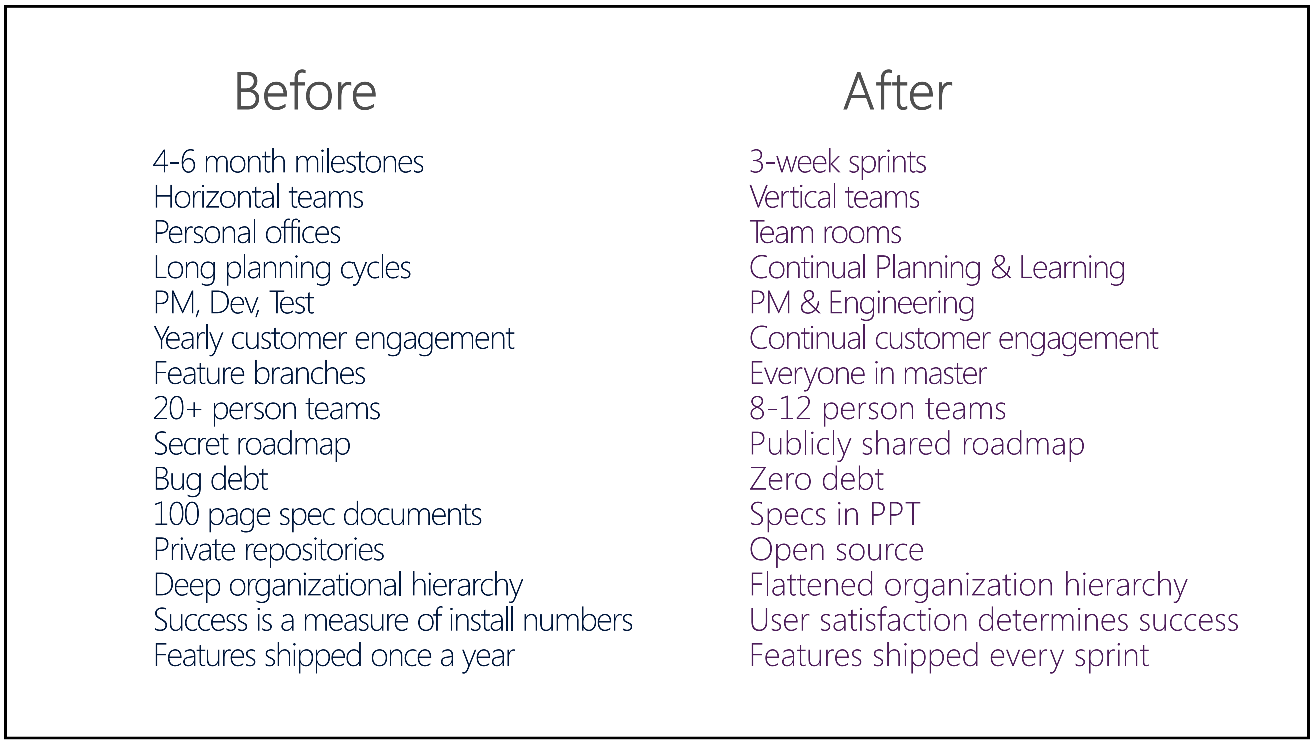
Manually triggered
by Hao Jiang (VSNC)
8/9/2018 12:16 PM

Artifacts
VSO.Release.CI
VSO.Release.CI_M138_20180809.2
releases/M138

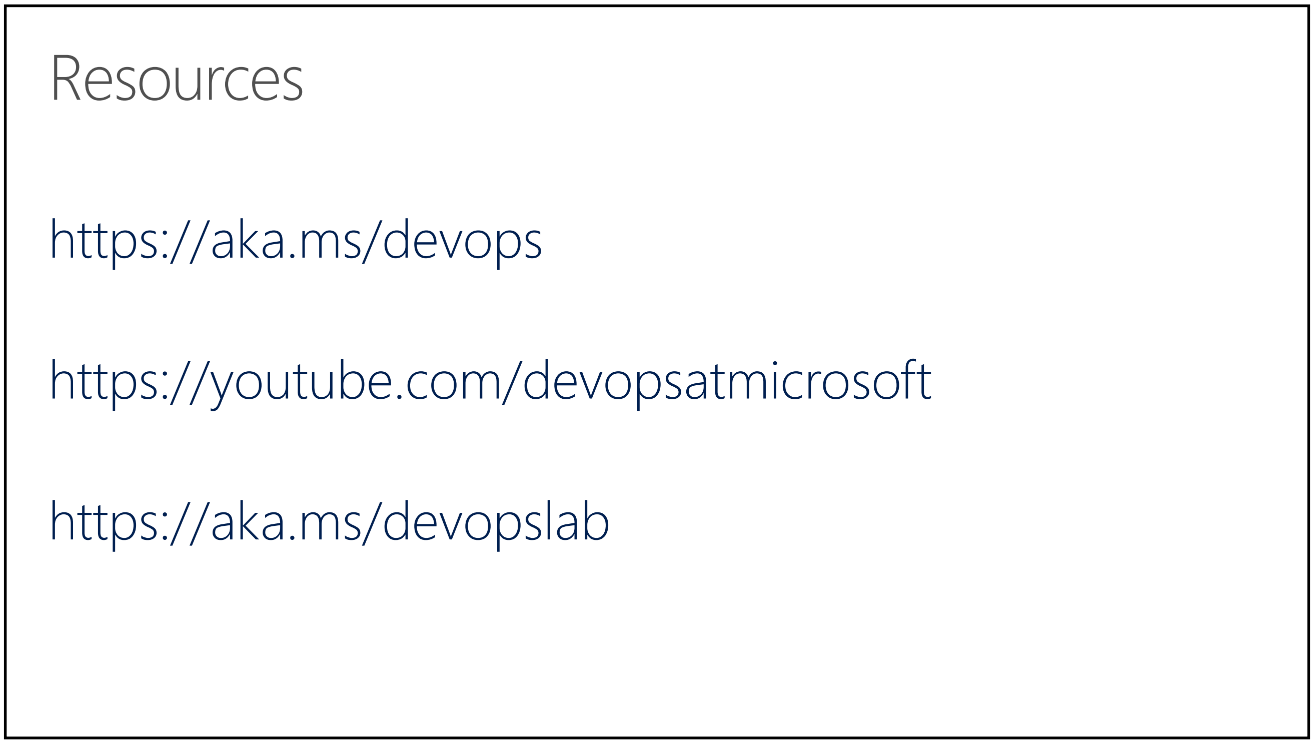
Environment	Status	Detail
Ring 0 - tfs-ppe-su1	Succeeded	on 8/9/2018 4:37 PM
Ring 0 - tfs-wcus-0	Succeeded	on 8/9/2018 3:44 PM
Ring 1 - tfs-sbr-1	In progress for 1h	Phase 2/4, 3/4 tasks, Prod Update 43:35
Ring 2	Not	
Ring 0 - tfs-eus2-su2	Succeeded	on 8/9/2018 4:29 PM
Ring 0 - tfs-pfcusc	Succeeded	on 8/9/2018 3:36 PM
Ring 1 - su6	In progress for 23m	Phase 2/4, 3/4 tasks, Prod Update 16:35

Wrap-Up

Microsoft



Before	After
4-6 month milestones	3-week sprints
Horizontal teams	Vertical teams
Personal offices	Team rooms
Long planning cycles	Continual Planning & Learning
PM, Dev, Test	PM & Engineering
Yearly customer engagement	Continual customer engagement
Feature branches	Everyone in master
20+ person teams	8-12 person teams
Secret roadmap	Publicly shared roadmap
Bug debt	Zero debt
100 page spec documents	Specs in PPT
Private repositories	Open source
Deep organizational hierarchy	Flattened organization hierarchy
Success is a measure of install numbers	User satisfaction determines success
Features shipped once a year	Features shipped every sprint

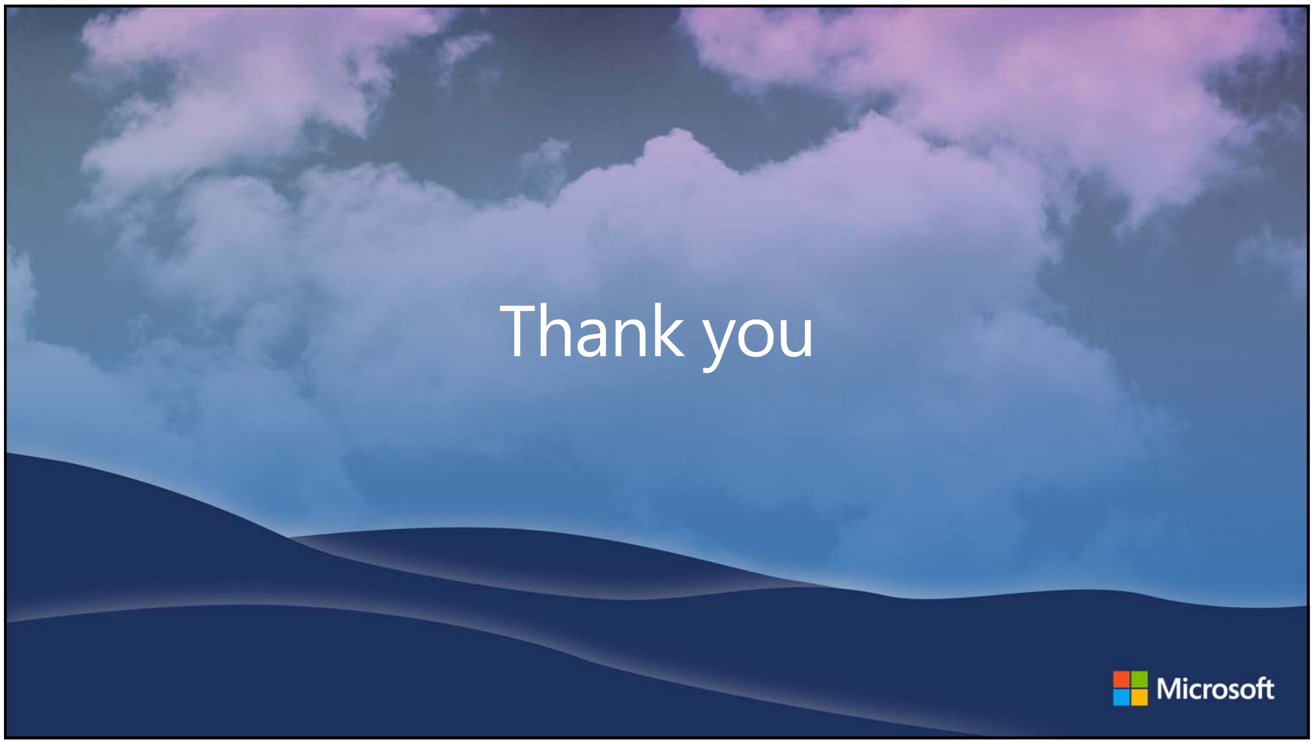


Resources

https://aka.ms/devops

https://youtube.com/devopsatmicrosoft

https://aka.ms/devopslab



Thank you