

Workshop

State of Washington

3 session agenda

Session #1 - learning session

↪ Overview, methods, and goals

Session #2 - small group work session

↪ Product vision, roadmap review, and decision points

Sessions #3 - small group work session

↪ QASP review + RFQ Drafting!

WA HHS Coalition milestones

Jan - Mar 2022	Apr - Jun 2022	Apr-June 2022	Jul-Sept 2022
<ul style="list-style-type: none">Platform and Product 1 RFPs are written + shared w/ CMSIdentify and prioritize questions that need to be answered to enable best collaboration with vendorsStart prep for M&O recompete	<ul style="list-style-type: none">Hiring and staffing internal positionsRFPs are releasedLicences + environments start getting set up as possibleConduct research + collaboration sessions to answer questions + prep for vendorsStart drafting M&O recomplete RFP	<ul style="list-style-type: none">Vendors evaluated + selectedContracts approved by CMS + executedPrep for next DP beginsM&O Recompete RFP is outBegin business + policy processes to create streamlined application	<ul style="list-style-type: none">Vendors start workNext DP is submittedM&O vendor(s) selected

Agile Contract Format

Background & purpose

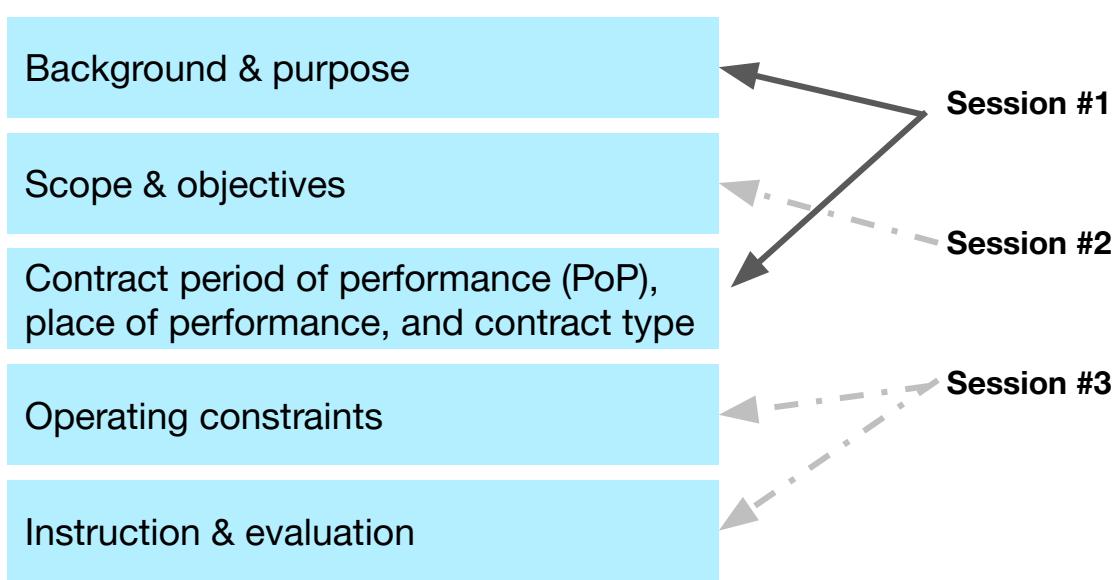
Scope & objectives

Contract period of performance (PoP),
place of performance, and contract type

Operating constraints

Instruction & evaluation

Agile Contract Format



Session #1 agenda



Welcome!

We are honored to work with you.

18F



Office of Financial Management
Better information. Better decisions. Better government. Better Washington.

**18F is a digital services consultancy for
government, inside the U.S. government.**

(We're federal employees.)

We help agencies
build, buy, and
share the best
digital government
services available.



Your 18F team



Nina Mak
Product Management Lead



Greg Walker
Consulting Engineer + Project lead



Randy Hart
Acquisition Lead



Austin Hernandez
Senior Product Designer



Amy Ashida
Principal Consultant



Erica Vosseller
Account Manager

Who is here today?

Write in the chat:

- Your name
- Your division and program
- Your role in this meeting
- One thing that makes you proud to work for your organization

Session norms

During this time we're aiming to build shared understanding of goals, ways of working, and paths forward. How will we as a group behave to make that possible?

- Take space + make space for others
- Be comfortable with silence
- Put challenges in the middle of the table for consideration by all
- Interrupt with questions
- We'll maintain a parking lot of things to come back to
-
-
-
-

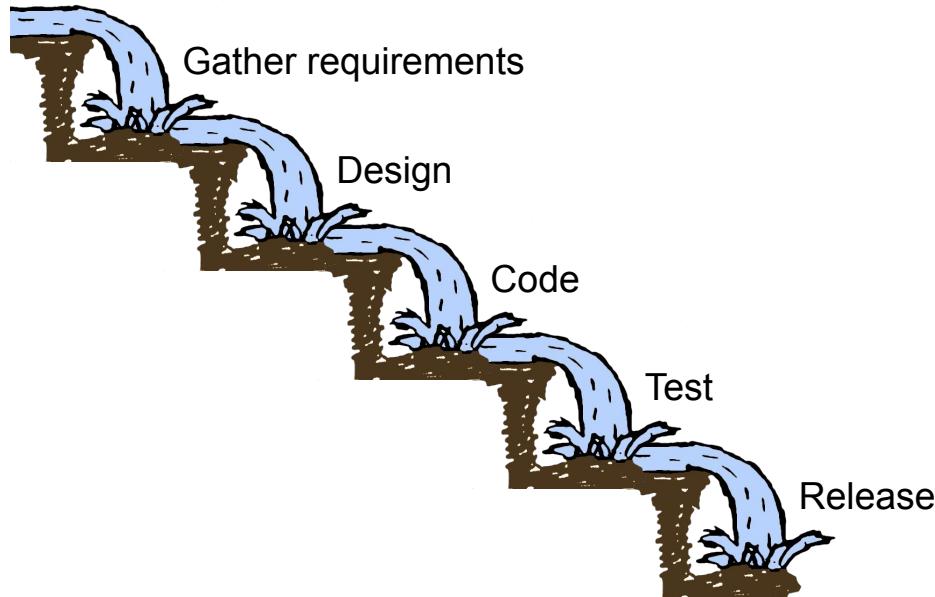
A quick poll to get us started

Agile & Agile misconceptions



Waterfall development

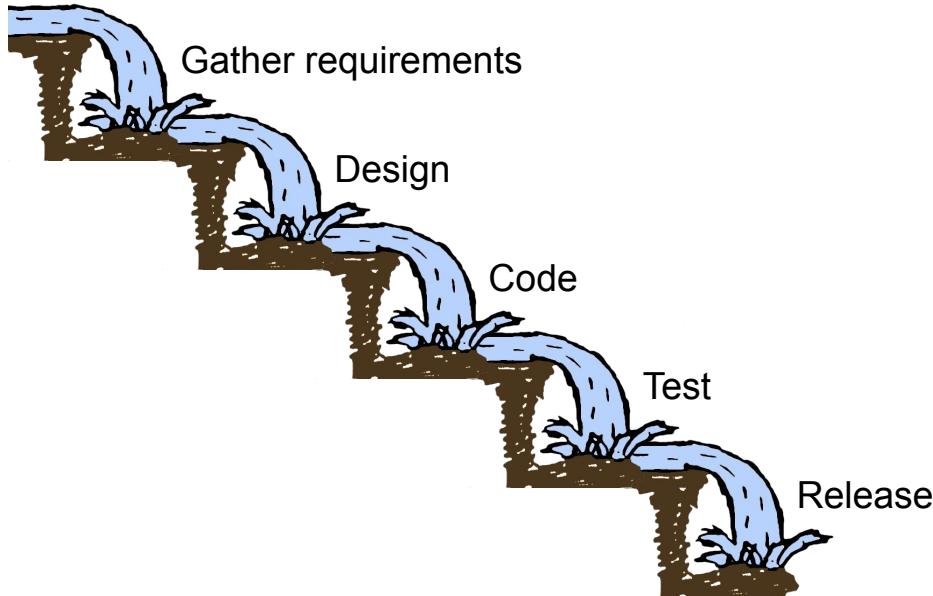
Plan



- Introduced by Winston Royce in 1970, *Managing the Development of Large Software Systems*

Waterfall development

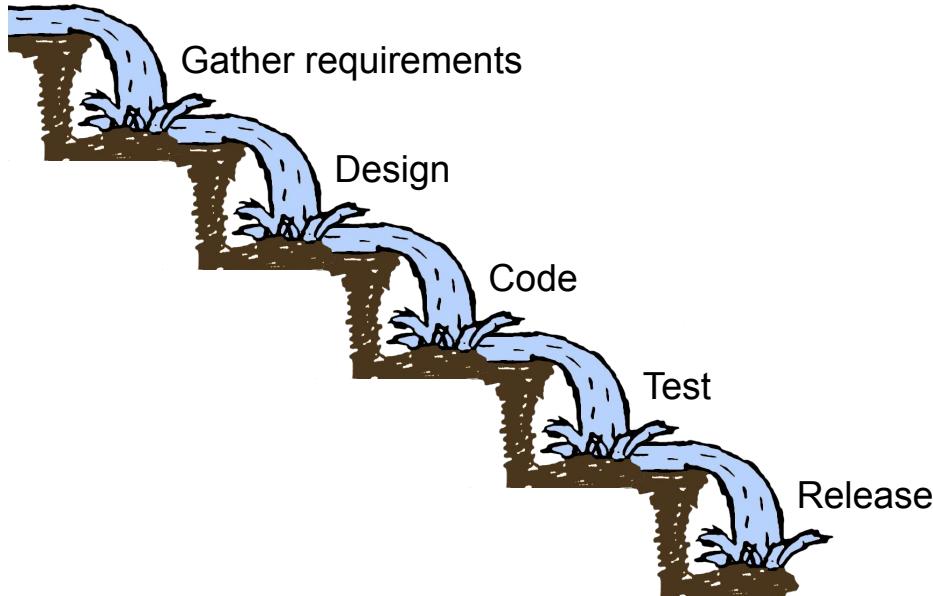
Plan



- Introduced by Winston Royce in 1970, *Managing the Development of Large Software Systems*
- He showed a variation of this diagram. It's right after the 4th paragraph of the paper. Just before the 5th paragraph.

Waterfall development

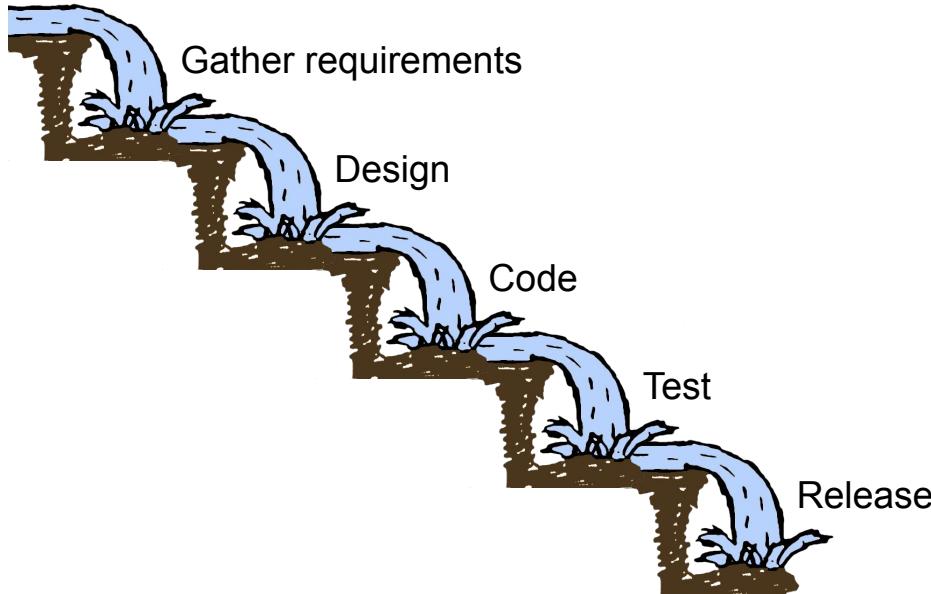
Plan



- Introduced by Winston Royce in 1970, *Managing the Development of Large Software Systems*
- He showed a variation of this diagram. It's right after the 4th paragraph of the paper. Just before the 5th paragraph.
- The 5th paragraph begins by saying it won't work.

Waterfall development

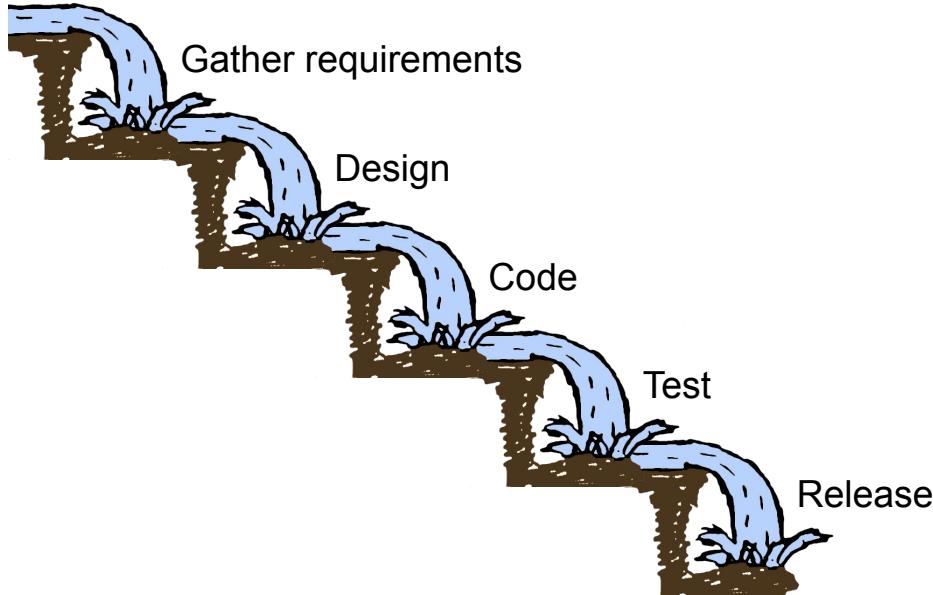
Plan



- Introduced by Winston Royce in 1970, *Managing the Development of Large Software Systems*
- He showed a variation of this diagram. It's right after the 4th paragraph of the paper. Just before the 5th paragraph.
- The 5th paragraph begins by saying it won't work.
- We never should have done this.

Waterfall development

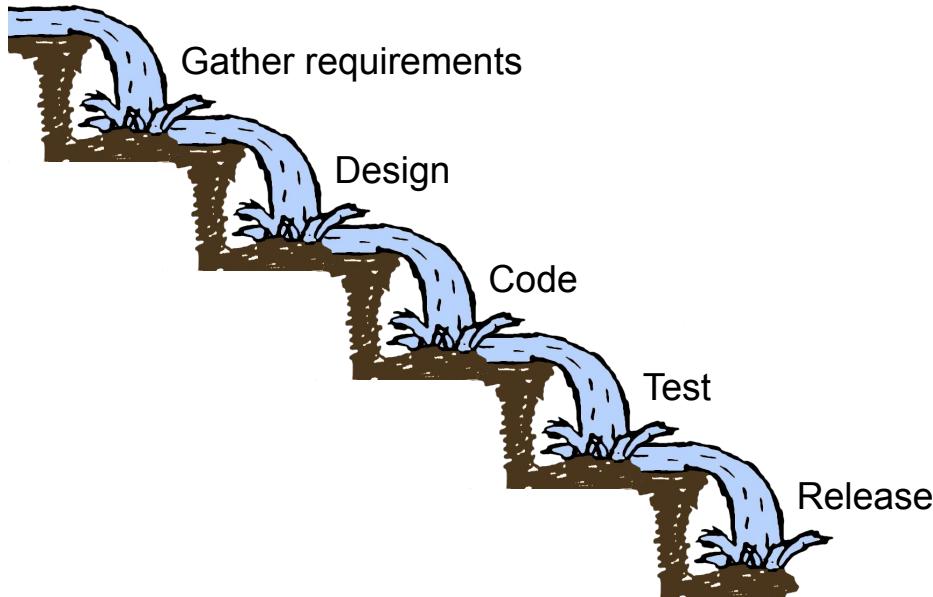
Plan



- Introduced by Winston Royce in 1970, *Managing the Development of Large Software Systems*
- He showed a variation of this diagram. It's right after the 4th paragraph of the paper. Just before the 5th paragraph.
- The 5th paragraph begins by saying it won't work.
- We never should have done this.
- The guy who identified it said so.

Waterfall development

Plan



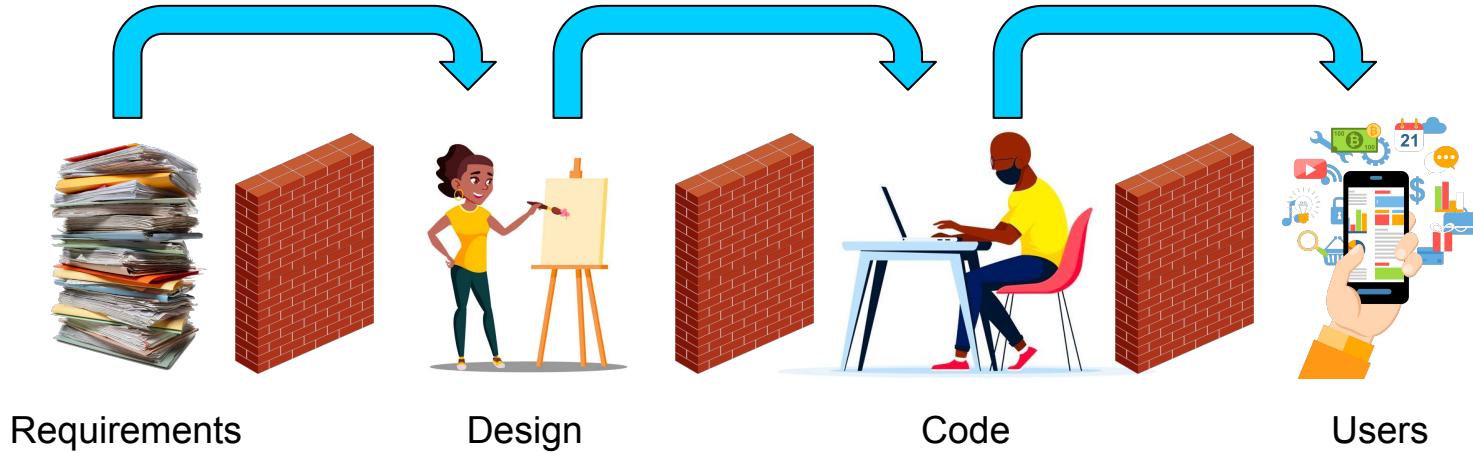
- Introduced by Winston Royce in 1970, *Managing the Development of Large Software Systems*
- He showed a variation of this diagram. It's right after the 4th paragraph of the paper. Just before the 5th paragraph.
- The 5th paragraph begins by saying it won't work.
- We never should have done this.
- The guy who identified it said so.
- He said we should be iterative, but we basically ignored him until 2001.

Agile is not only a way of working, it's a way of thinking and acting, it's our aspiration to build products faster, in a collaborative manner and, with the Customer on the center of our thoughts.”

- *João Baptista Leite, CEO, Unicre – Instituição Financeira de Crédito, S.A.*

Agile vs waterfall

Waterfall and the walls



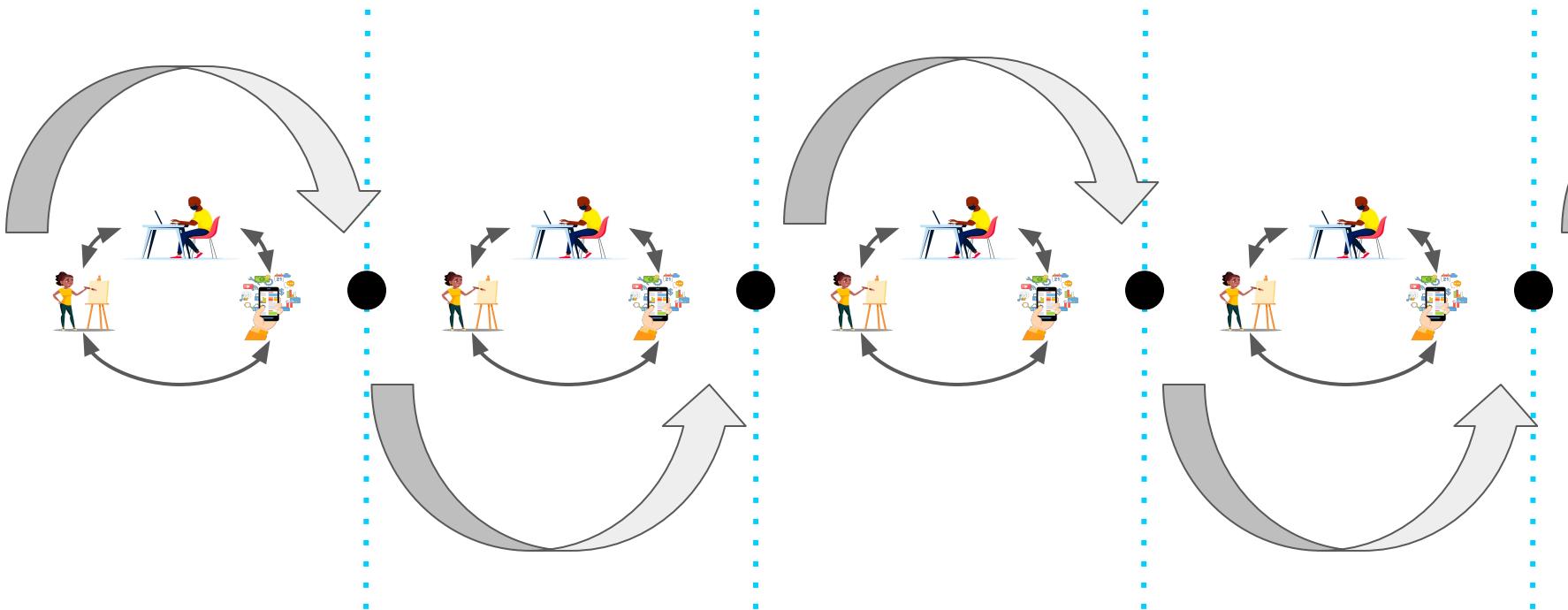
Agile comes in like a wrecking ball



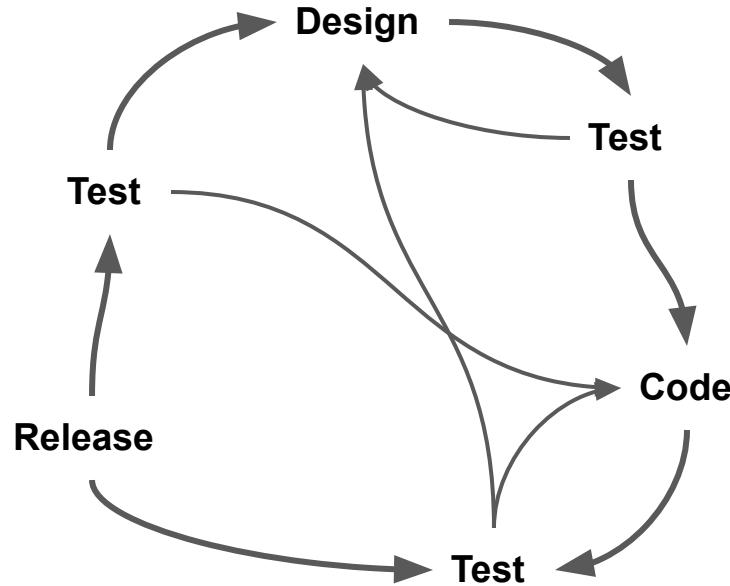
Agile comes in like a wrecking ball



Agile keeps stakeholders in the loop



Agile is full of loops



Agile misconceptions

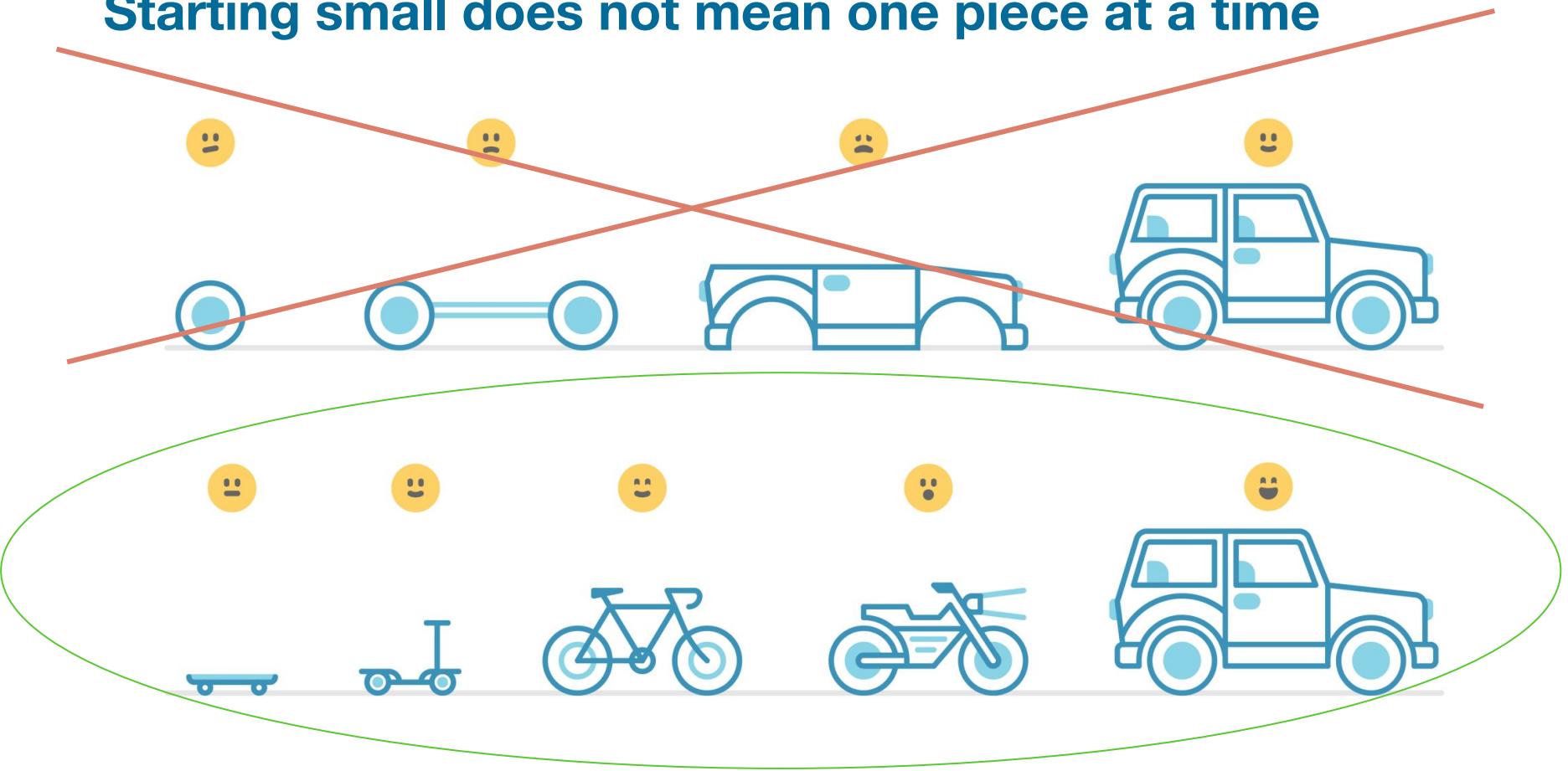
Common assumptions made about agile and what we mean



Common misconceptions:

- Starting small means one piece at a time
- Agile means less planning
- Agile means less/no documentation
- Agile is fast/less work

Starting small does not mean one piece at a time



Agile does not mean no/less planning

1. Learn about users



2. Make prototypes
and test them



3. Build a small version
of the real thing



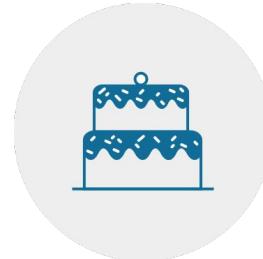
4. Get feedback from
real users



5. Use feedback to
adjust and improve



6. Rinse and repeat!



Agile does not mean no/less documentation

- Transparency of work and team capacity
- Shared vision/goals
- Clear idea of what has happened and what needs to be done
- Less rework and duplication of work
- Technical and research findings for future teams

The image shows a Kanban board with three columns: "Backlog / Next Up", "Current Sprint", and "In Progress / Doing". The "Backlog / Next Up" column contains two items:

- "Update design guidance based on user feedback" (status: In Progress)
- "Writing guidelines" subpage: Revise and add the content writing scorecard - @malaika18f (Consider for 1/14) (status: To Do)

The "Current Sprint" and "In Progress / Doing" columns are currently empty.

Below the Kanban board is a large screenshot of a README document for a project named "Project Name".

README

Project Name

Tock line: #1418
GitHub staffing issue: <https://github.com/18F/staffing/issues/791>
Timeframe: 03/26/21 - 05/10/21
Project background for Phase One:
Based on our initial research, we concur with the 10x pitch author that: "tribal

stently across and no examples of ated specifically to is creation of an N people in user g, to ensure that implementation of on plans as required where to start.

Authentication

The application uses Okta to provide authentication. Users must register for an Okta account and be added to the application user group. Users must also sign up for Multi Factor Authentication (MFA) with Okta. They have the option of using voice call, email, sms text, or a software-based authenticator (such as Google Authenticator or Okta Authenticator) to provide one-time passwords or multi factor credentials.

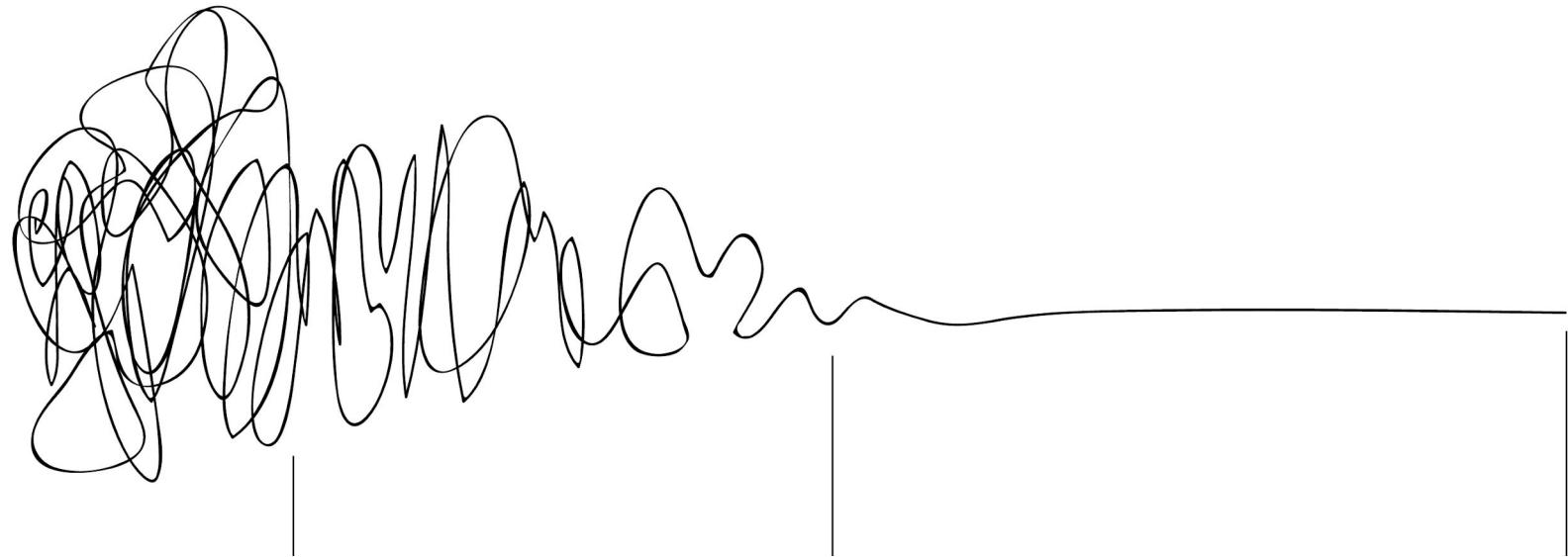
Authentication is a multi-step process. The front-end uses the `@okta/okta-auth-js` node module to create an `oktaClient` object that is used to interact with Okta. First, the `oktaClient sign-in` method sends the user's username and password to Okta. Okta returns a response with different potential statuses. Error statuses include `LOCKED_OUT` or `PASSWORD_EXPIRED`. The user is instructed on how to handle these situations.

If the user has not yet set up a multi-factor option, Okta will return a status of `MFA_ENROLL`. The user will then be walked through the steps to set up their second factor. If they choose to use email, Okta will use their EUA email. If they choose CALL or SMS, the user will have the option of supplying a phone number for Okta to use.

If the user has already set up a second factor, Okta will return a status of `MFA_REQUIRED`. The transaction includes an array of factors that the user has set up. That factor has a function named `verify` that will send the one-time password to the user when it is called. The user is then redirected to the page where they will enter the one-time password that they received from Okta.

Once the user enters the one-time password, `oktaClient` resumes the previous transaction, which has a `verify` function that takes the one-time password. If this one-time password is valid, Okta returns a transaction with the status 'SUCCESS' and a session token. The front-end then uses the `oktaClient` to get the access token using the session token and a state token generated by the front-end. Okta returns the same state token and the access tokens. That Okta token is finally exchanged

Agile is not (necessarily) fast/less work



More discovery work upfront

To avoid rework and costly oversights

Product ownership

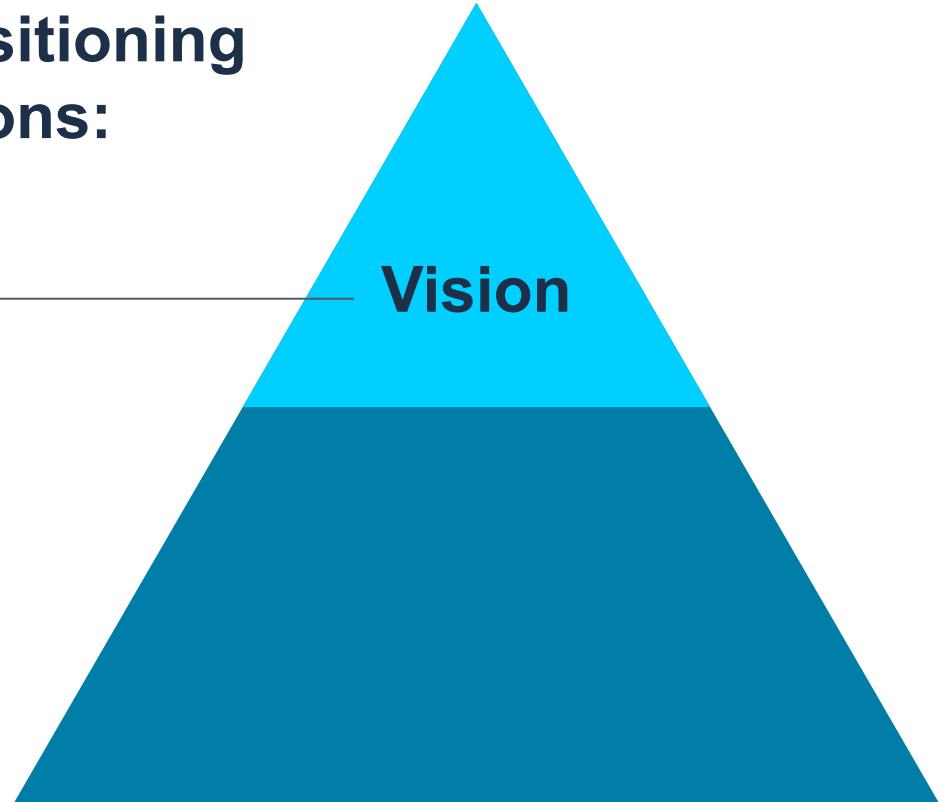
Moving toward successful outcomes (not output)



**Who owns the vision? The strategy?
The State of Washington**

**Product uses that positioning
to answer two questions:**

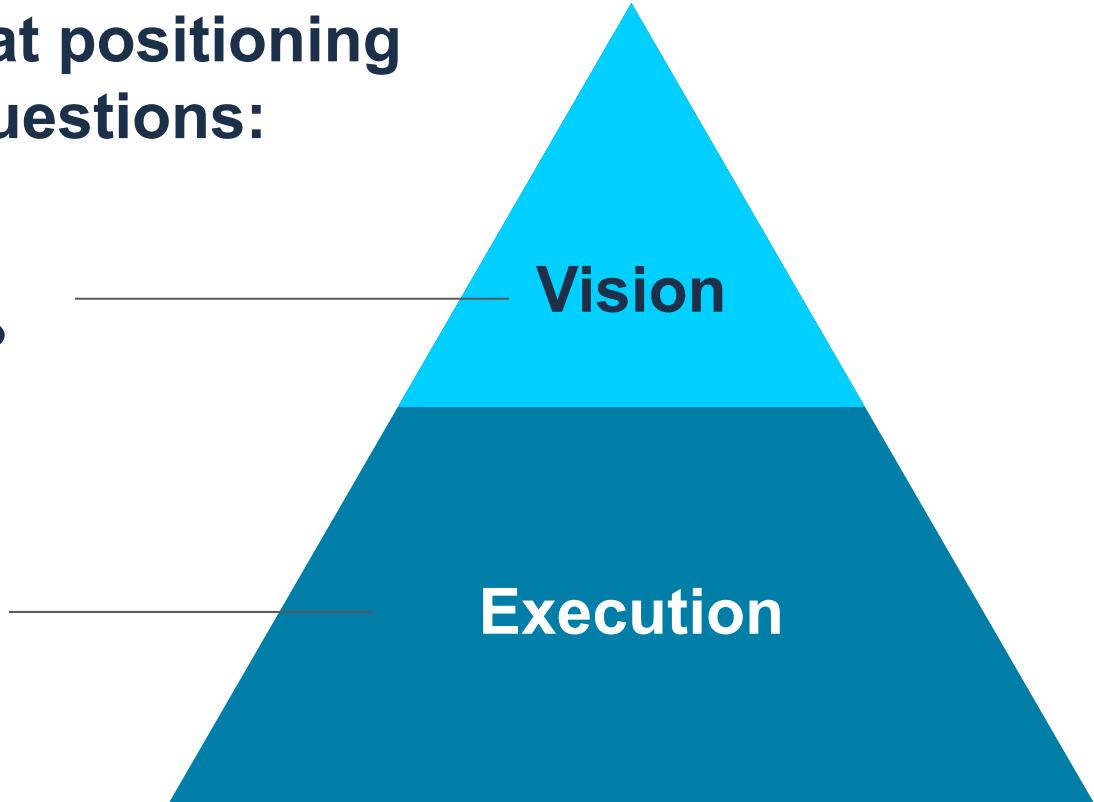
- 1. What should
we do? Why?**



**Product uses that positioning
to answer two questions:**

- 1. What should
we do? Why?**

- 2. How do we
get there?**



Product owners collaborate with team members and stakeholders to create a **shared vision**, then communicate it internally and externally.

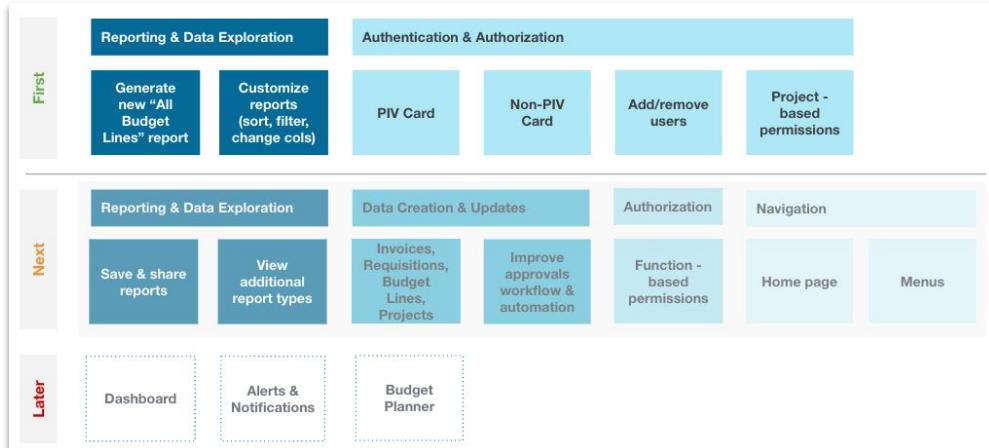
From the vision, product owners create strategy, roadmap, and feature definition of a product.

Product roadmaps are a powerful tool for product teams charting out a strategic plan

Roadmap Board

[Menu](#)

Tasks	Members	Status	May
COMPY-2049 Create tutorial video		In Progress	<div style="width: 100%;">COMPY-2049: Create tutorial video</div>
COMPY-2048 Update project plan		To Do	<div style="width: 0%;">COMPY-2048: Update project plan</div>
> Later			<div style="width: 100%; background-color: #e0e0e0;">Later</div>
- Blocker			<div style="width: 100%; background-color: #e0e0e0;">Blocker</div>
COMPY-1123 Time zones		Designing IN PROGRESS	<div style="width: 100%;">COMPY-1123: Time zones</div>
COMPY-1234 Basic batch actions		Developing	<div style="width: 100%;">COMPY-1234: Basic batch a...</div>
COMPY-1328 Backup and rollback		To Do	<div style="width: 0%;">COMPY-1328: Backup and rollba...</div>
COMPY-1482 Multiple checklists		To Do	<div style="width: 0%;">COMPY-1482: Multiple checklist...</div>
COMPY-1581 New billing		Designing IN PROGRESS	<div style="width: 0%;">COMPY-1581: New billing</div>
+ Add Task			
COMPY-1579 Automated email marketing		Designing IN PROGRESS	<div style="width: 100%;">COMPY-1579: Automated email marketing</div>
COMPY-1421 PPM improvements		Developing	<div style="width: 0%;">COMPY-1421: PPM improvements</div>
COMPY-1029 Localization: German		To Do	<div style="width: 0%;">COMPY-1029: Localization: German</div>
COMPY-1564 Search engine optimization		To Do	<div style="width: 0%;">COMPY-1564: Search engine optimiz...</div>
+ Add Group	+ Add Task		



Product roadmaps communicate priorities and sequence

DENVER CONNECTED PEDESTRIAN ROADMAP

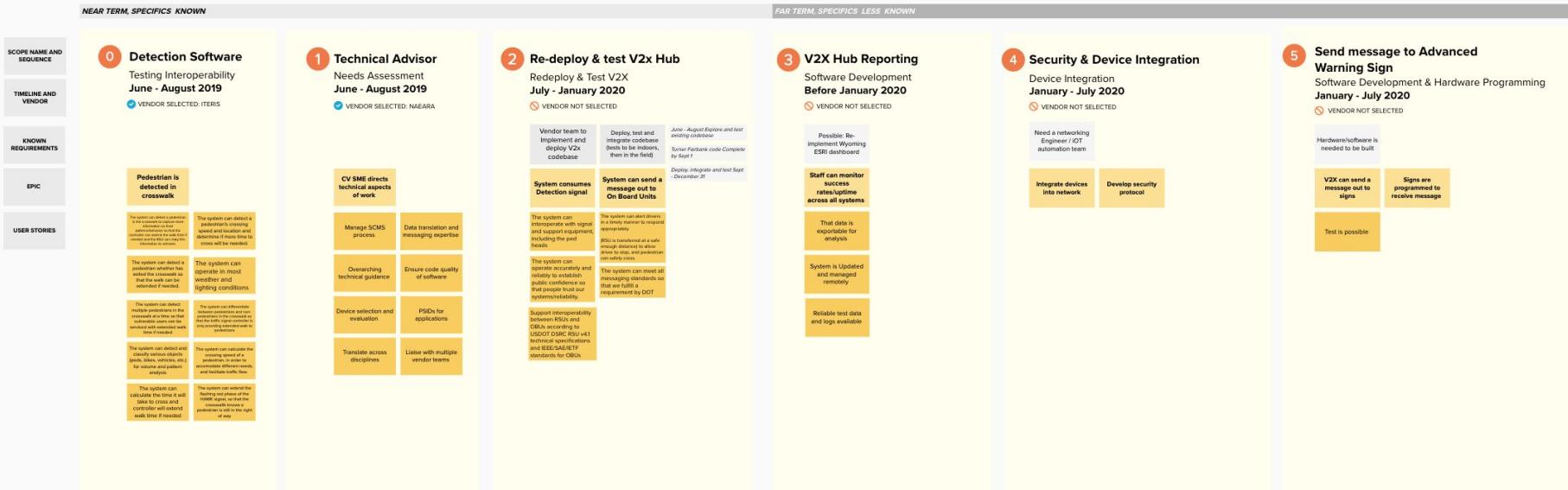
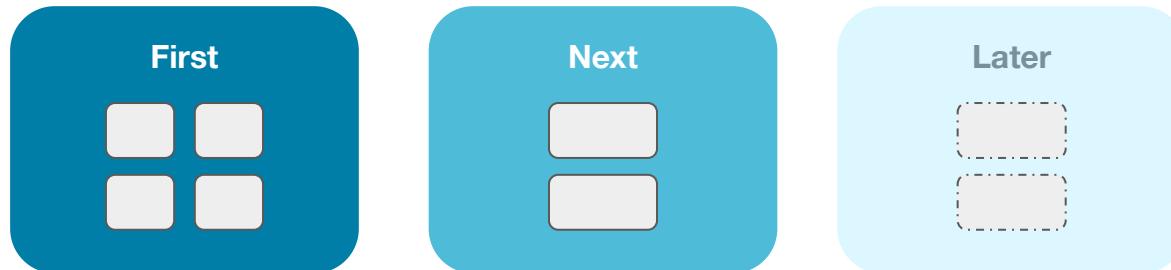


Image 9: IE&E Modernization Roadmap



Status Tracker Roadmap



**The roadmap is a bridge between
your strategic vision and your
backlog of work**

**Product owners protect the
project team, scope, and what
gets built.**

Product owners protect the project team, scope, and what gets built.

They should say “no” often, and leadership should give them the space to do so.

Design and execution matter



Product owners own four questions:

**1. What's the problem we're solving?
(and why does it matter?)**

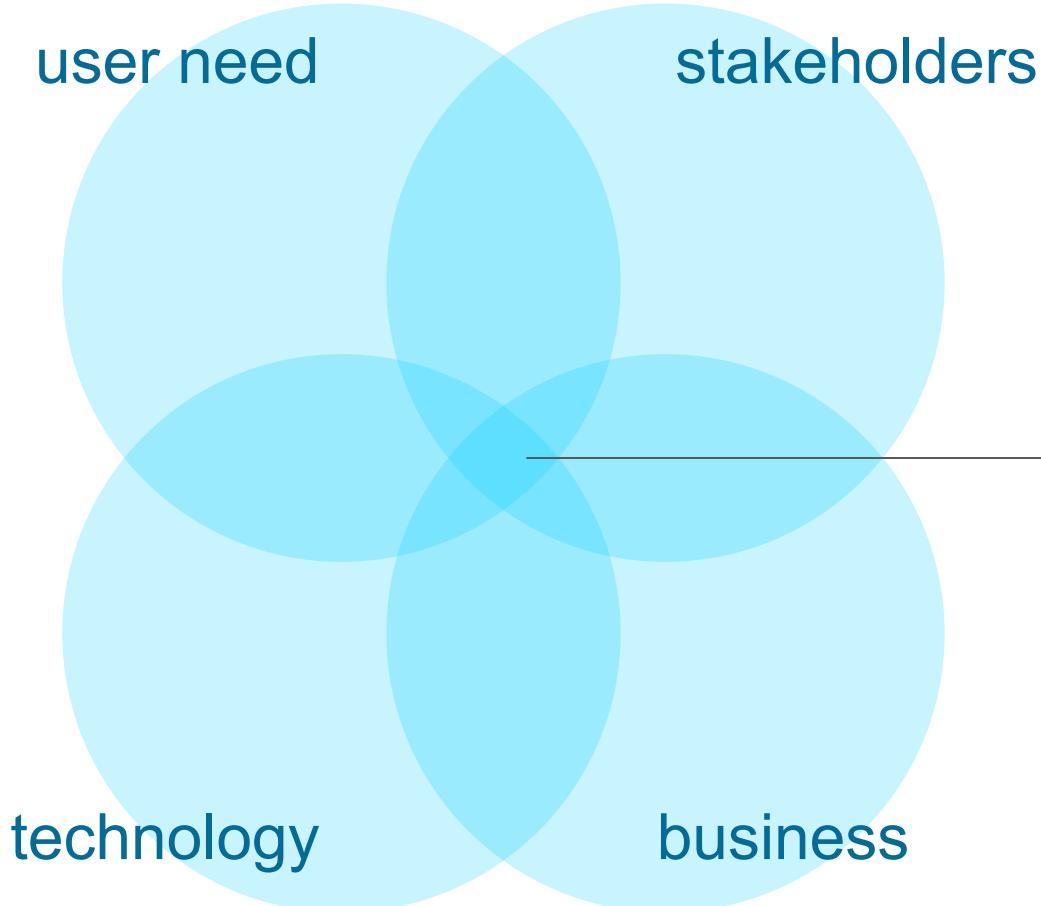
1. What's the problem we're solving?
(and why does it matter?)
2. What will our impact be?

1. What's the problem we're solving?
(and why does it matter?)
2. What will our impact be?
3. **How are we doing it?**

1. What's the problem we're solving?
(and why does it matter?)
2. What will our impact be?
3. How are we doing it?
- 4. Is our solution good?**

Product teams have to be able to provide good answers to all of these questions.

Product teams have to be able to provide good answers to all of these questions. When the answer is “we don’t know,” they need to have a concrete plan to get to an answer.



Product owners sit at the intersection of design and research, development, business, policy, and politics.

A word of warning:
product ownership is a
major commitment.

Treating product owners as “other duties as assigned” is not setting the product owner, or the effort, up for success.

Product owners' days are filled with

Planning and strategizing

- Defining outcomes
- Breaking down the work
- Resolving ambiguity
- Driving decisions & alignment
- Prioritizing
- Balancing

Executing and delivering

- Refining the backlog
- Writing user stories
- Clearing blockers for the team
- Reviewing work

Maintaining and retrospecting

- Monitoring usage
- Soliciting feedback
- Prioritizing fixes

Meeting with users

Communicating with stakeholders

Attending usability sessions

Human-centered design

Putting people at the center of all problem solving



**Human-centered design helps us
design for actual humans who
will use what we're making.**

Who are the people you are designing for?

_____ **who** _____

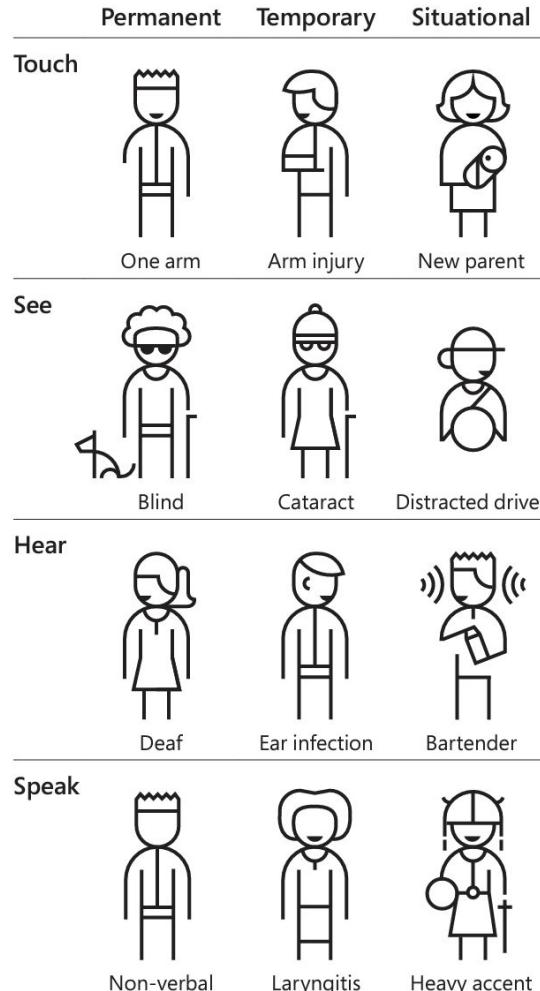
Ex.

A person who receives benefits.



Microsoft's inclusive design principles:

- **Solve for one, extend to many.**
- **All humans grow and adapt to the world around them.**
- **Exclusion happens when we solve problems using our own biases.**
- **Include and learn from people with a range of perspectives.**



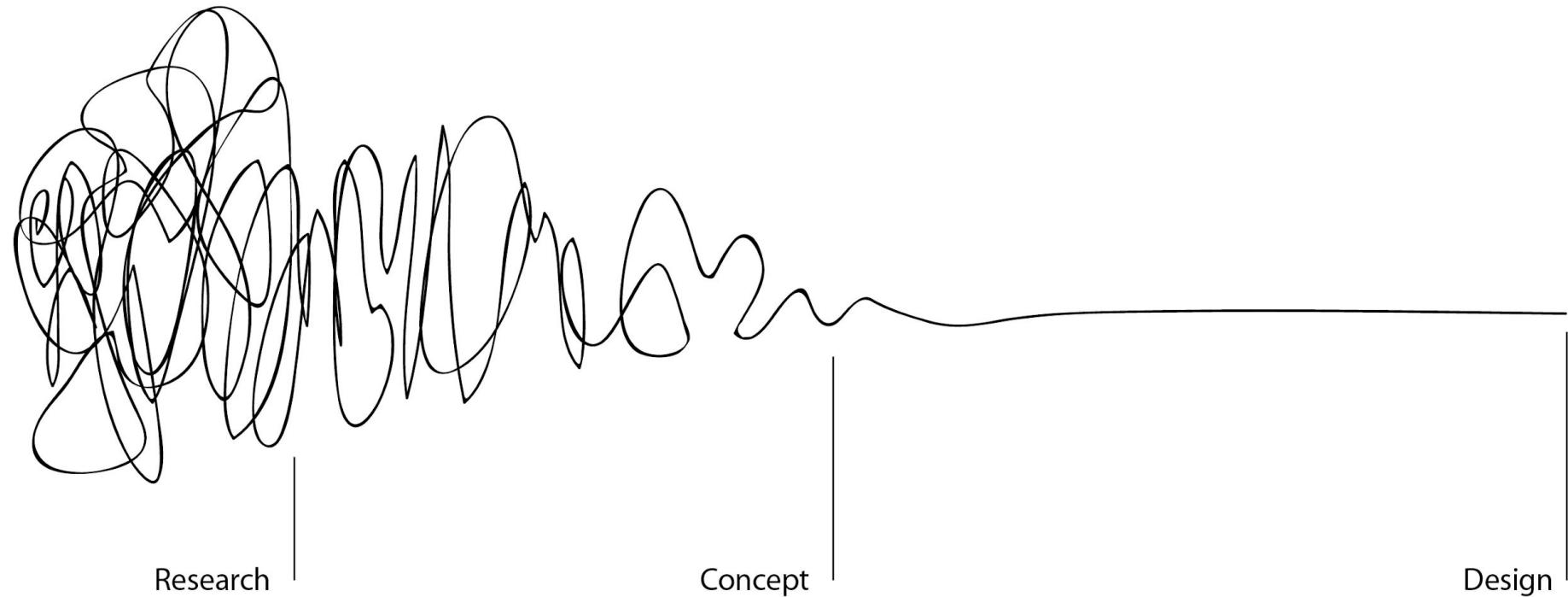
The Persona Spectrum

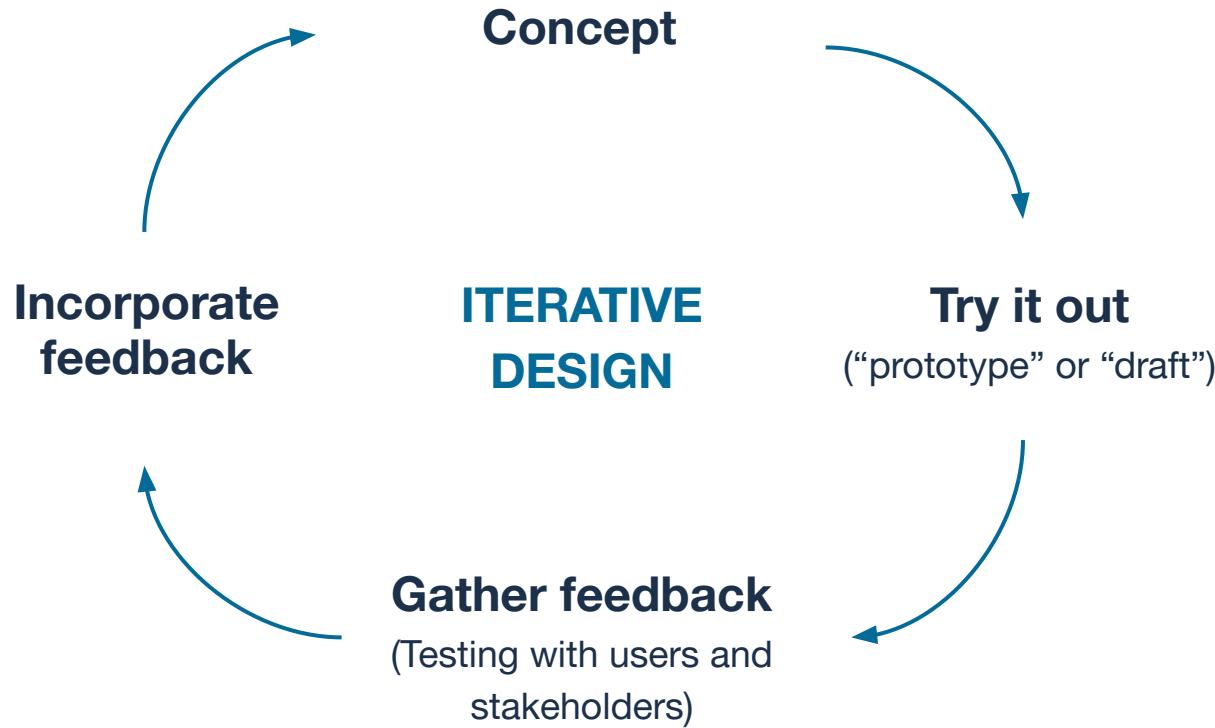
We use the Persona Spectrum to understand related mismatches and motivations across a spectrum of permanent, temporary, and situational scenarios. It's a quick tool to help foster empathy and to show how a solution scales to a broader audience.

**There's no one design process
but there are consistent patterns.**

Uncertainty / patterns / insights

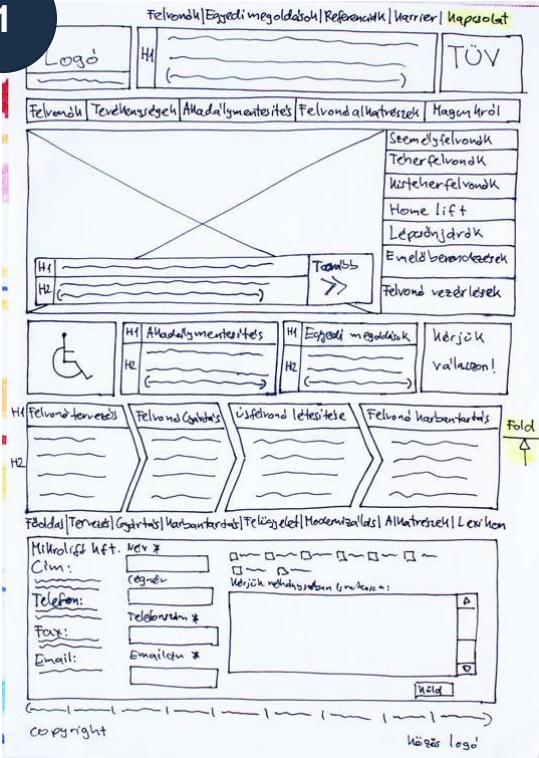
Clarity / Focus



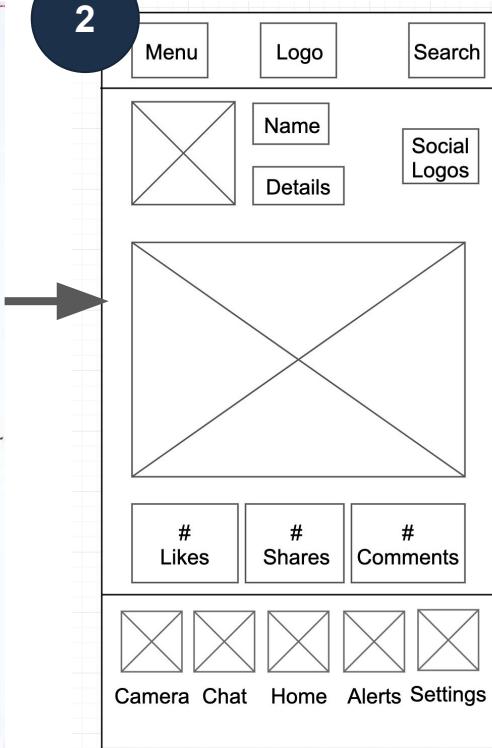


Trying things out doesn't need to be costly

1



2



3

website of the United States government

National Science Foundation

News Awards Education Multimedia Data Search Find funding

NSF funded the research that brought you iPhone screens

Learn more Apply for a grant

Featured

Latest news Most popular

NSF-FUNDED June 11, 2018 Inside a turtle's brain

NSF-FUNDED June 11, 2018 Coral responds to acid

NSF-FUNDED June 11, 2018 Comb-jellies shed light on deep sea life

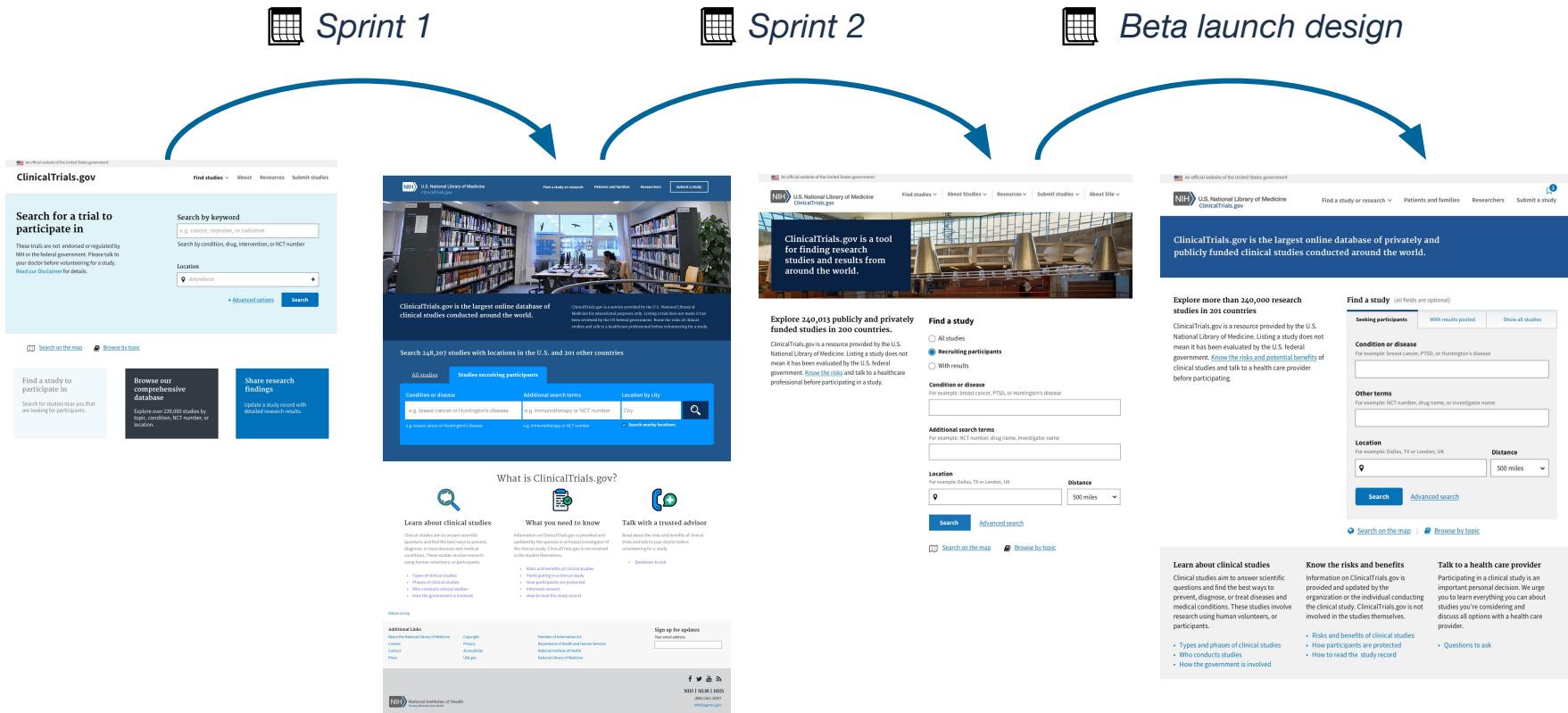
New dinosaur in Tanzania

A newly discovered titanosaurian dinosaur upends common theories about evolution.

READ FULL ARTICLE

Browse more news

Learn from people and incorporate feedback





Product Owner



Tech Lead



Project Manager



Business Analyst



Procurement Specialist

Everyone has a role to play



Research Lead



Note taker / Documenter



Interviewer/Moderator



Coordinator



Observer

Everyone has a role to play

**Good design
habits to start
practicing now**

-  **Do user research early and often**
-  **Rely on assumptions about your end-users**



**Explore
multiple approaches**

**✗ Double down
on a single
solution early**

-  **Define the outcomes you want**
-  **Stick to a predefined solution**

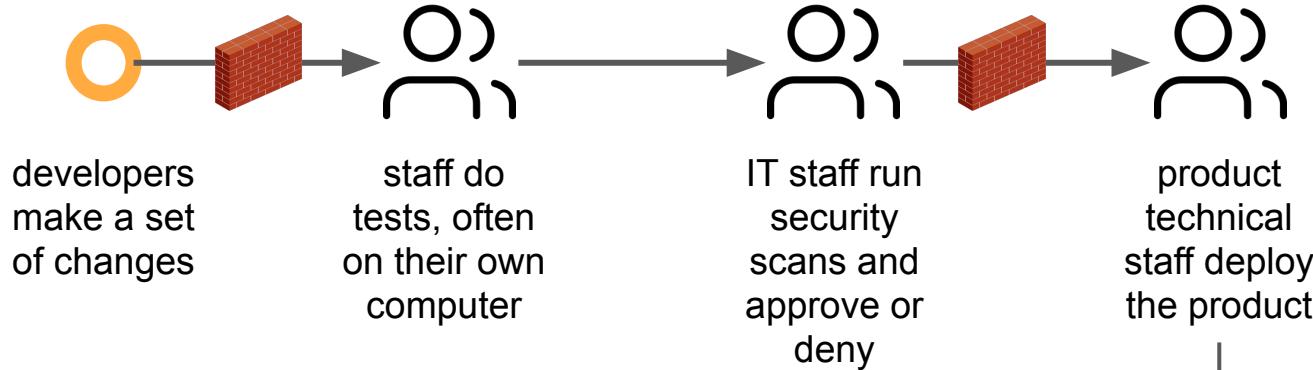
DevSecOps

Helping us build robust solutions that we can constantly test, integrate, and deploy

Letting people and computers do the things they're each good at

DevSecOps combines software development, security, and operations into one team, and automates as many of their processes as possible. People focus on the hard problems and let the computers handle the repetitive stuff.

A manual deployment process



```
Processes: 123 total, 2 running, 129 sleeping, 56 threads
Load average: 0.00 0.00 0.00 CPU usage: 1.0% user, 0.0% nice, 2.0% sys, 55.0%idle
SharedLibs: 3508K resident, 5760K data, 68K linkedin.
HeapPages: 1110M shared, 1110M private.
Memory: 2.1M wired, 3288M active, 750M inactive, 5340M used, 1024M free.
VM: 2308 voice, 1054M from 454M virtual, 1792M(81%) pagesize, 0(0) pageouts.
Network: 0(0) rx, 0(0) tx, 0(0) dropped, 0(0) errors, 0(0) collisions.
DiskIO: 229509/3409M read, 418661/7724M written.

PID COMMAND XCPU TIME #TH #W #R #P#R #MREG #P#V#T PSH#D RSIZE+
1477 top 13.9 00:01.38 1/1 0 24 33 1488K+ 244K 1598K+
1480 MailComp_138 0.0 00:00:00 0 0 14 25 141K+ 141K 141K
1463 bash 0.0 00:00:00 0 1 0 17 25 256K 955K 955K
1462 login 0.0 00:00:00 1 1 0 23 62 616K 320K 244K
1460 MailComp_138 0.0 00:00:00 0 0 14 25 141K+ 141K 141K
1465 Cathode 0.0 00:01.88 5 2 127 267 280+ 984+ 659+
1466 MailComp_138 0.0 00:00:00 0 0 14 25 141K+ 141K 141K
1468 wufileLookd 0.0 00:00:00 2 0 37 46 238K 420K 420K
1461 dcspd 0.0 00:00:01 2 0 42 40 738K 319K 216K
1462 MailComp_138 0.0 00:00:00 0 0 14 25 141K+ 141K 141K
1294 Google Chrome 0.0 00:02.07 1 1 93 97 48K 89K 89K
1132 MailComp_138 0.0 00:00:00 0 0 14 25 141K+ 141K 141K
1164 MailComp_138 0.0 00:01.57 8 0 178 220 149K 420K 420K
1165 MailComp_138 0.0 00:01.19 4 1 93 346 19K 87K 87K
1161 Gd 0.0 00:00:00 1 0 14 23 104K 244K 434K
```

The DevSecOps culture



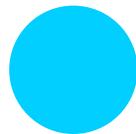
Open communications and
knowledge sharing
enhance trust within the team



Continuous feedback
results in tighter processes
that get better and better



Emphasizes the entire
system, not just one piece,
department, or interest



Experimentation lets us
learn from successes and
failures, constantly

DevSecOps covers three broad areas

1

Testing - Making sure things work as expected

2

Continuous Integration - Regularly making sure everything still comes together correctly

3

Continuous Delivery - Regularly publishing the latest version of the product for users to use

1/ Testing

Does it do what it's supposed to? Do the things that worked yesterday still work today?

Testing is about making sure the application does what it's supposed to and catching bugs as early as possible.

Testing the product

1

Assert what the features of the application should do:

```
hello-app
assert screen-output =
"Hello!"
```

Testing the product

1

Assert what the features of the application should do:

```
hello-app
assert screen-output =
"Hello!"
```

2

Write some code to implement those features:

```
puts "Hello!"
```

Testing the product

1

Assert what the features of the application should do:

```
hello-app
assert screen-output =
"Hello!"
```

2

Write some code to implement those features:

```
puts "Hello!"
```

3

Execute the tests to make sure the application does what we expect:

```
Finished in 0.3s, 1 assertion
0 failures, 0 errors
```

2/ Continuous Integration

Making sure the different parts go together, each step of the way

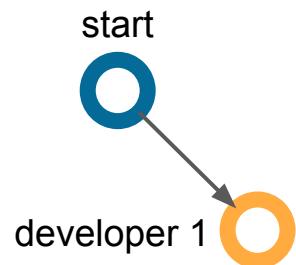
Continuous integration is taking each change, as it happens, and checking whether it breaks the product. This way the team knows quickly, as soon as a change is made, rather than days or weeks later.

From many, one

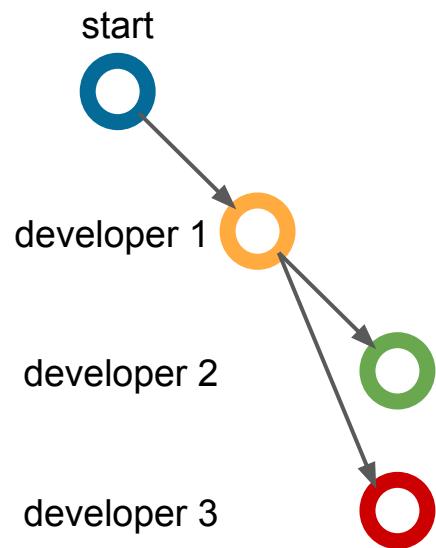
start



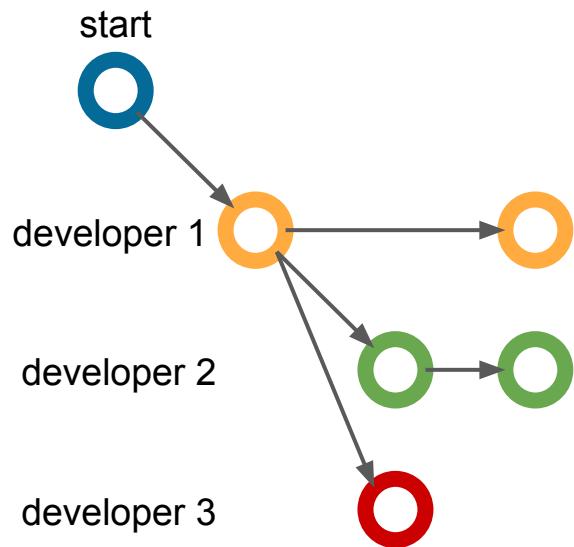
From many, one



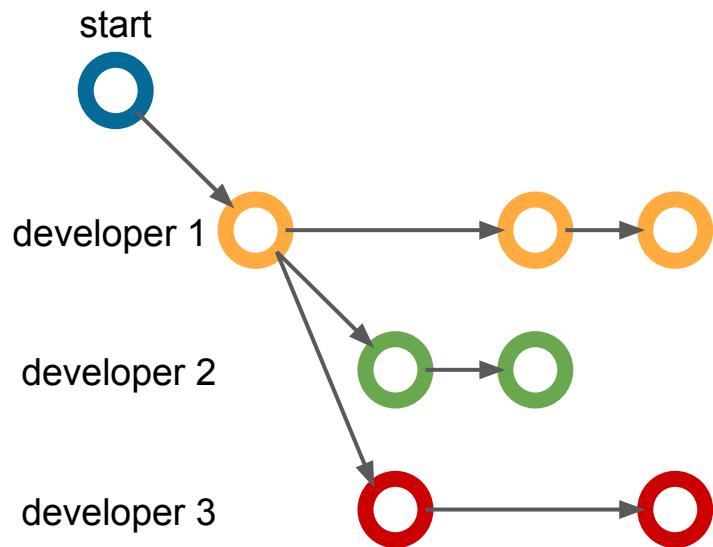
From many, one



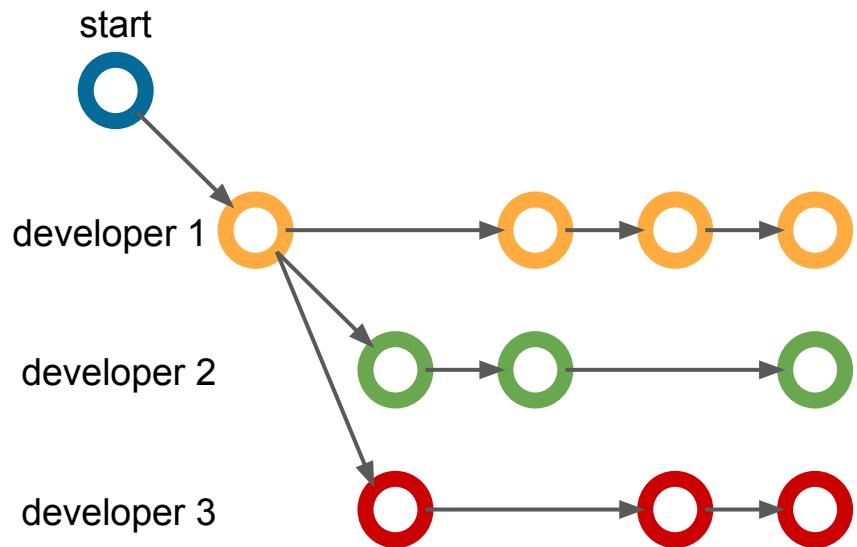
From many, one



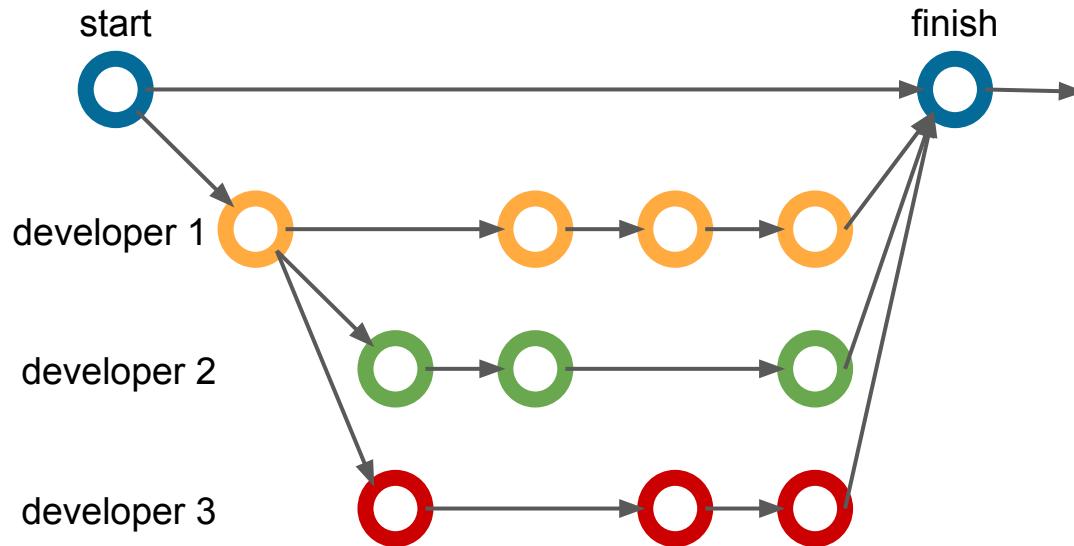
From many, one



From many, one

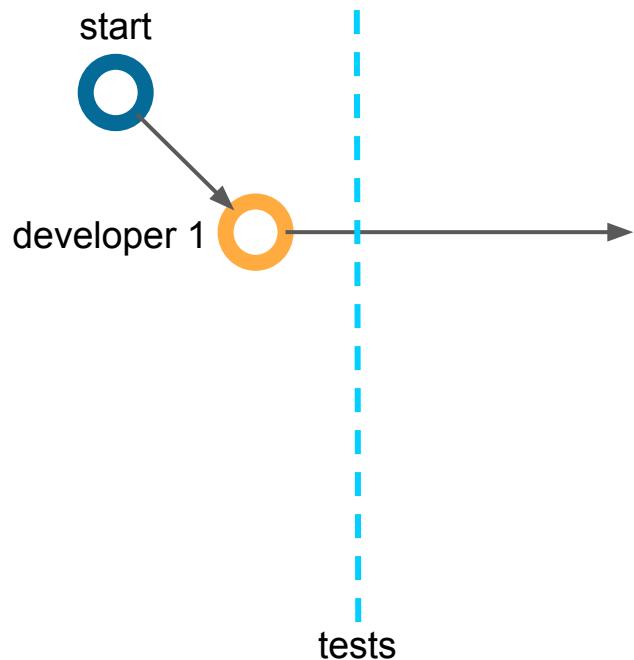


From many, one

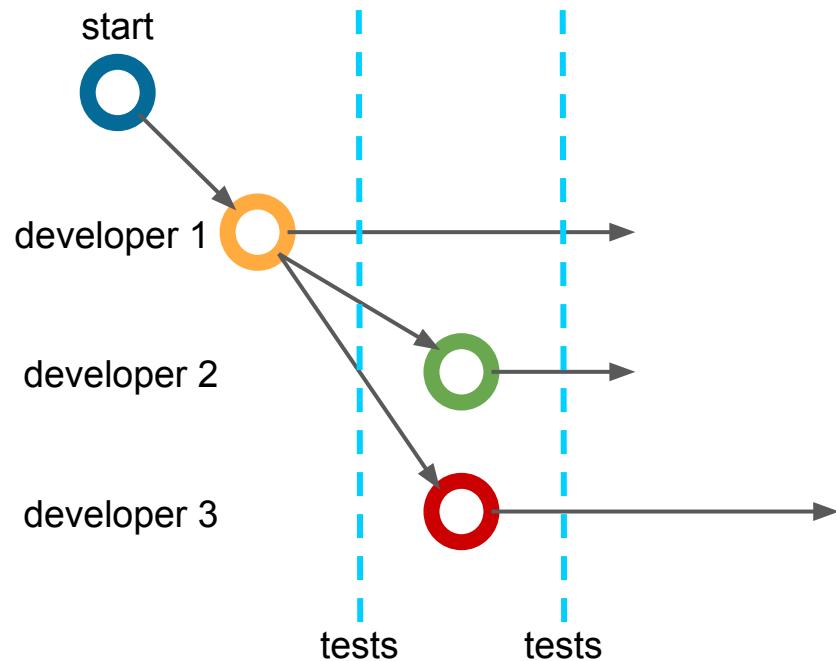


That looks like a
disaster about to
happen...

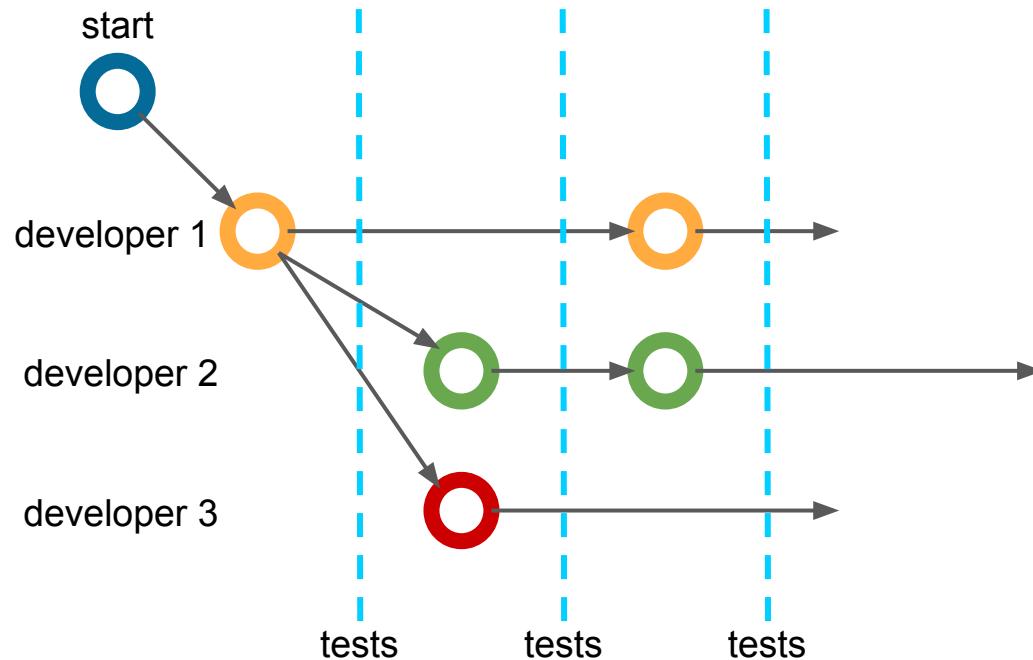
Everything is okay...



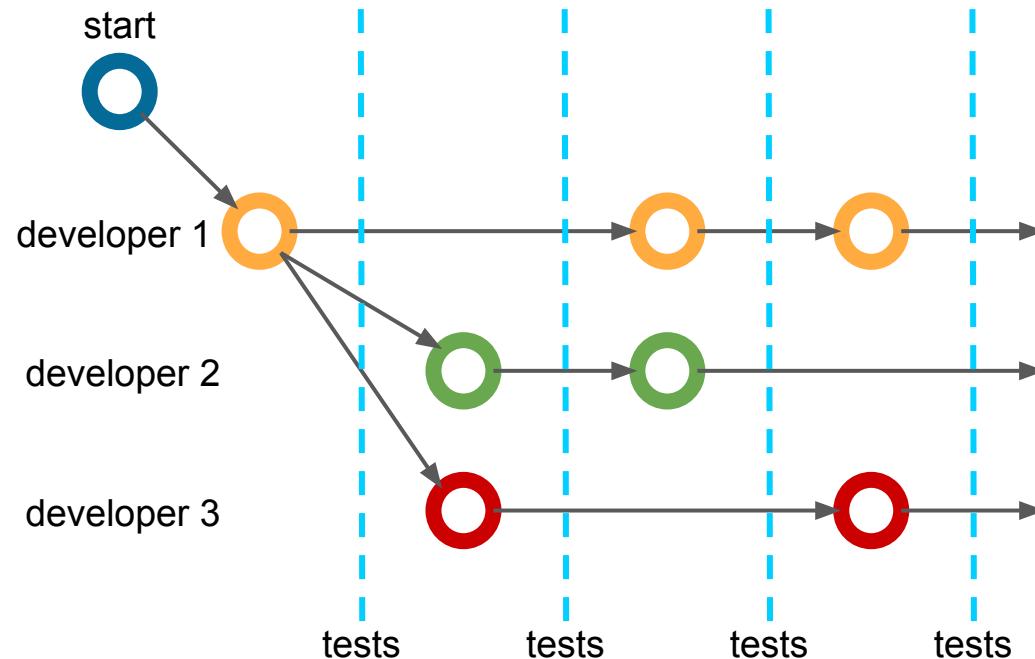
...and it's still okay...



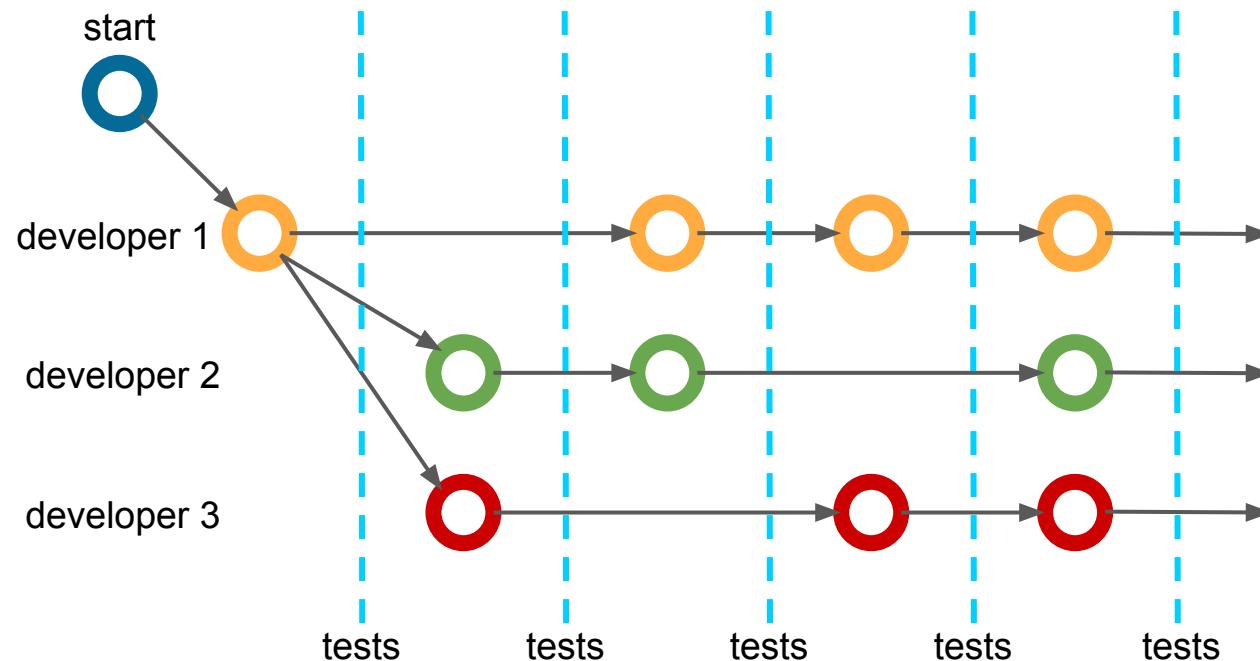
...and it's still okay...



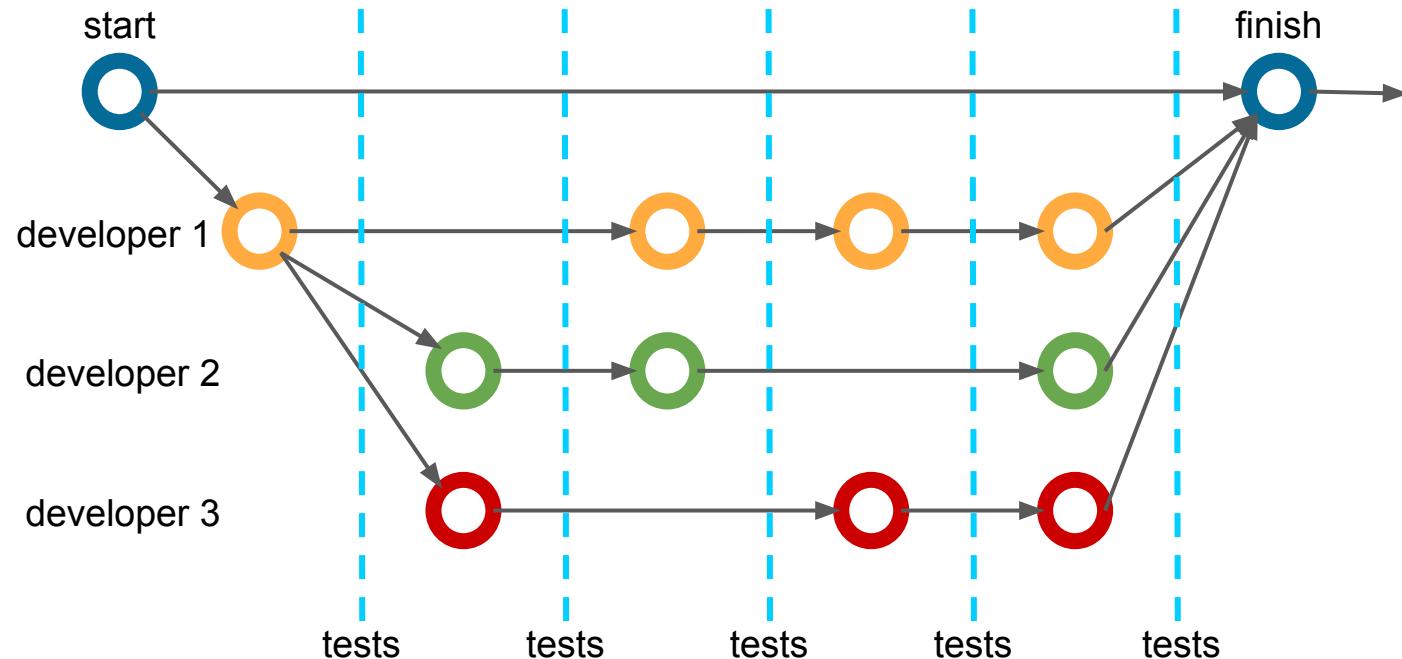
...and it's still okay...



...and it's still okay...



...and everything is good!

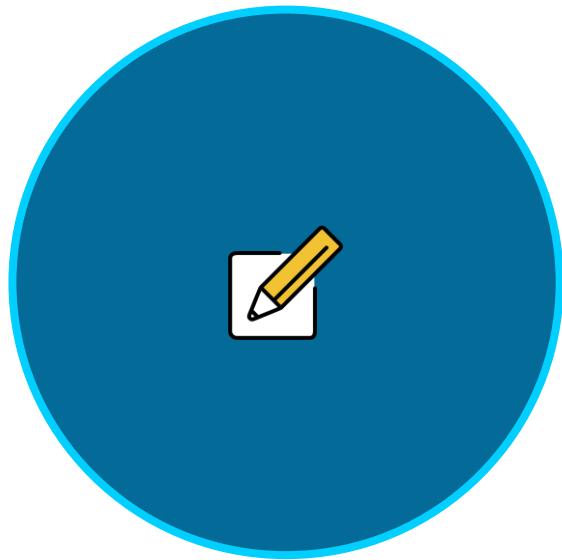


3/ Continuous Delivery

Making the new, updated product available as often as possible

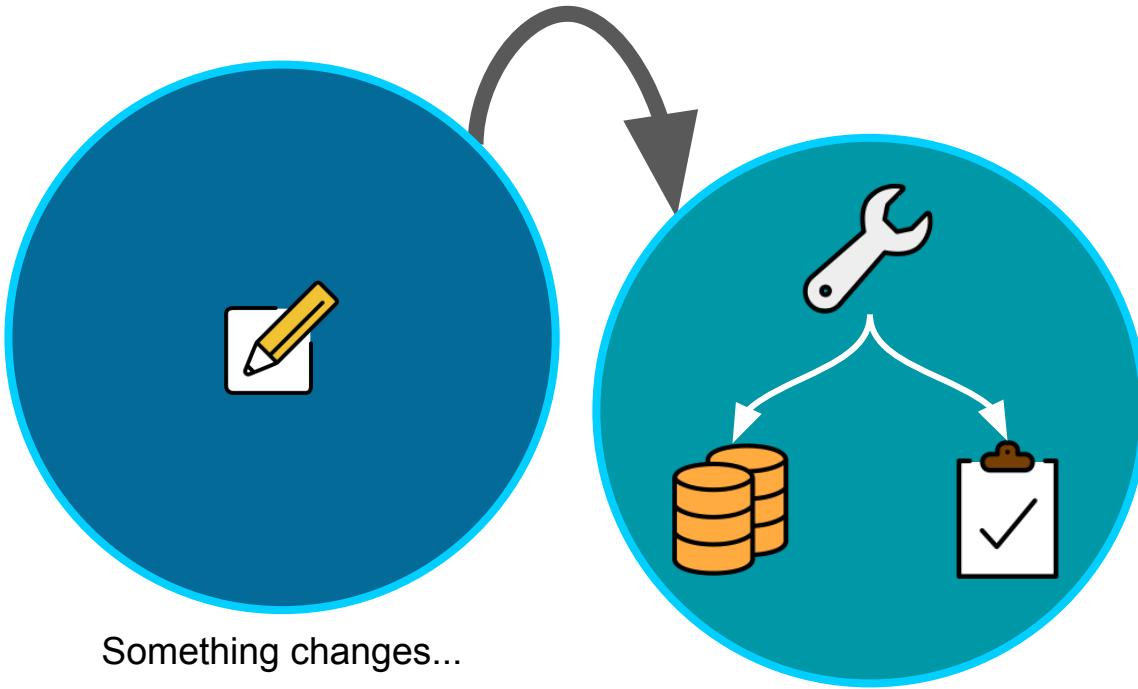
Continuous delivery is about deploying the most recent version of your product. That means every time some changes gets pulled into the app, the app gets deployed again.

The delivery



Something changes...

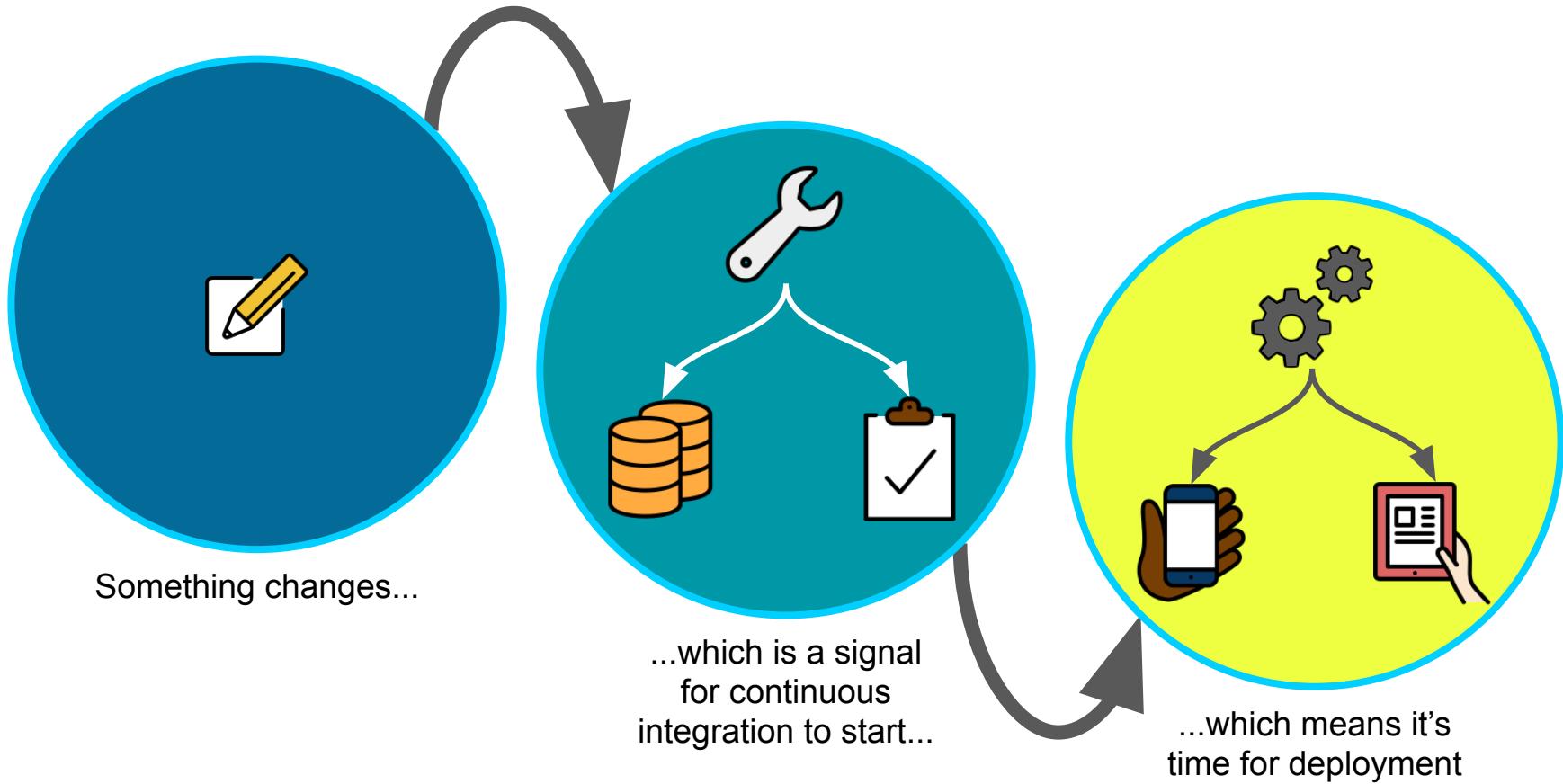
The delivery



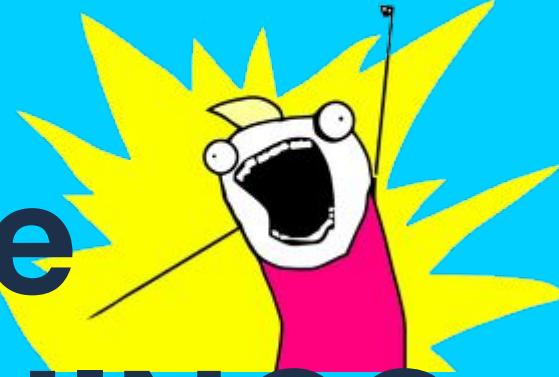
Something changes...

...which is a signal
for continuous
integration to start...

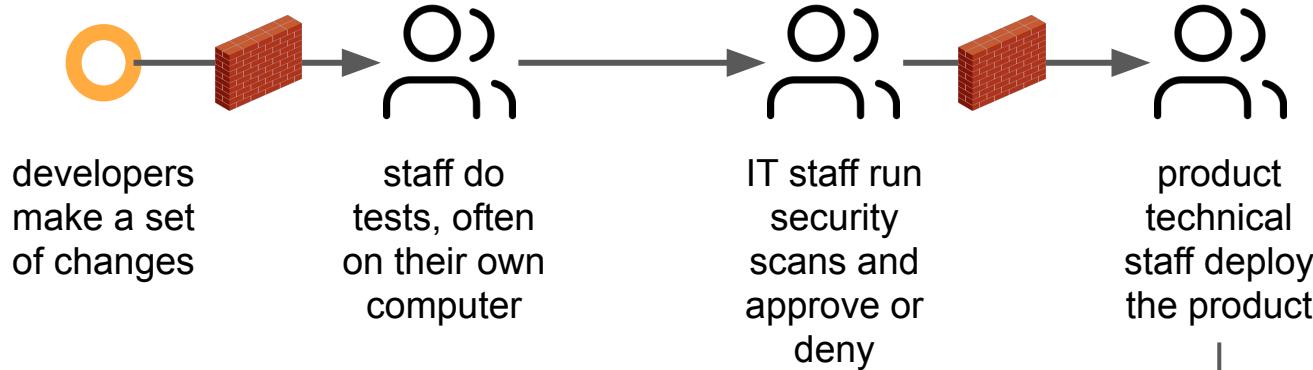
The delivery



4/ Automate ALL THE THINGS



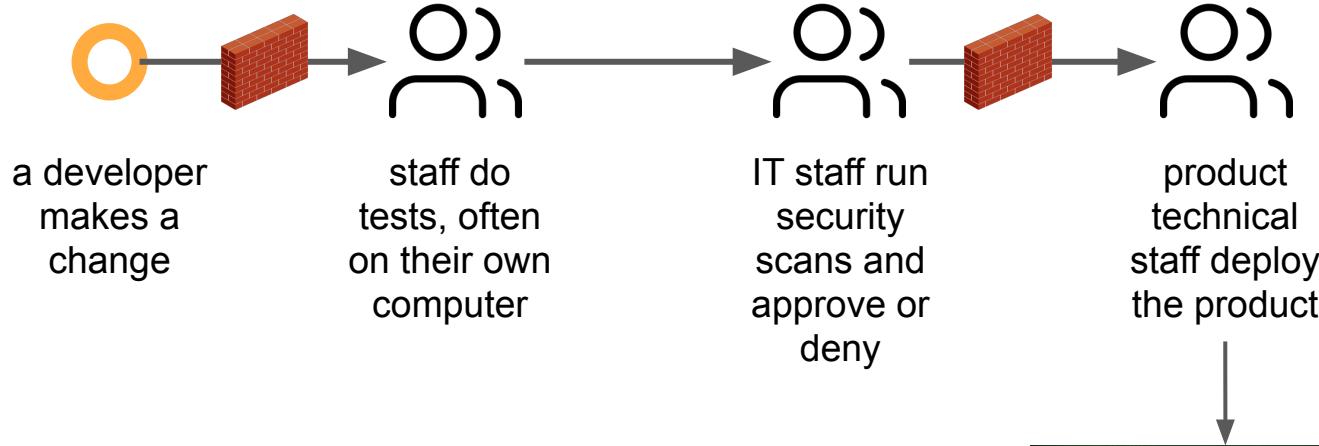
A manual deployment process



```
Processes: 123 total, 2 running, 120 sleeping, 828 threads
Load average: 0.00 0.00 0.00 CPU usage: user 14% sys 55.4% idle 85.6%
SharedLibs: 3508K resident, 5760K data, 681K linkedit.
Memory: 1.1G total, 1.1G used, 0 free. 1.1G available.
Physical memory wired, 3288M active, 785M inactive, 5349M used, 1024M free.
VM: 2308 voice, 1054M from 454M virtual, 1792M(81%) pagesizes, 0(0) pageouts.
Network: 0(0) receive, 0(0) transmit, 0(0) errors, 0(0) drops, 0(0) backlog.
DiskIO: 229509/3409M read, 418661/7724M written.

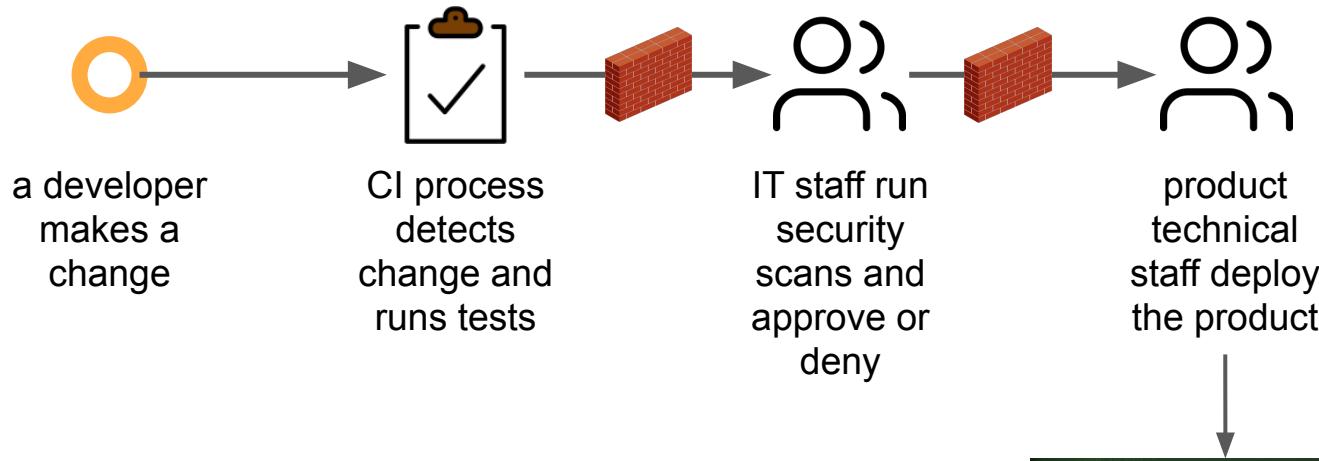
PID COMMAND XCPU TIME #TH #W #R #P#R #MREG #P#V#T PSH#D RSIZE+
1477 top 13.9 00:01.38 1/1 0 24 33 1488K+ 244K 1598K+
1480 MailScanner_138 0.0 00:00:00 1 0 14 25 141K+ 141K 141K
1463 bash 0.0 00:00:00 1 0 17 25 256K 955K 955K
1462 login 0.0 00:00:00 1 0 28 62 616K 320K 244K
1493 MailScanner_86 0.0 00:00:00 1 0 18 25 141K+ 141K 141K
1496 Cathode 0.0 00:01.88 5 2 127 267 284+ 984+ 659+
1481 httpd 0.0 00:00:00 1 0 25 46 238K 420K 420K
1482 wufileLookd 0.0 00:00:00 2 0 37 46 238K 420K 420K
1461 dcspd 0.0 00:00:01 2 0 42 40 738K 319K 216K
1483 Apache 0.0 00:00:00 4 0 14 25 141K+ 141K 141K
1294 Google Chrome 0.0 00:02.07 1 0 93 97 48K 89K 89K
1132 MailScanner_138 0.0 00:00:00 1 0 128 220 149K 149K 149K
1164 MailScanner_86 0.0 00:01.57 8 0 2 25 42M 42M 42M
1165 Apache 0.0 00:00:00 1 0 93 346 19K 87K 87K
1041 Gd 0.0 00:00:00 1 0 14 23 104K 244K 434K
```

A manual deployment process



PID	COMMAND	XCPU	TIME	#TH	WNO	NRPR	NRREG	RPRTV	PSHLD	RSSIZE
1477	top	13.9	00:01.38	1/1	0	24	33	1488K+244K	1598K+	1598K
1482	geditComp_138	0.0	00:00.00	1	0	14	14	148K+148K	148K	148K
1483	bash	0.0	00:00.00	0	0	17	25	256K+956K	956K	956K
1484	login	0.0	00:00.00	1	0	23	62	616K+320K	244K	244K
1485	geditComp_138	0.0	00:00.00	1	0	15	15	152K+152K	152K	152K
1486	Cathode	0.0	00:01.88	5	2	127	267	284K+984K	984K	984K
1487	geditComp_138	0.0	00:00.00	1	0	25	25	256K+956K	956K	956K
1488	wlclientd	0.0	00:00.00	2	0	37	46	238K+420K	420K	420K
1489	dcopd	0.0	00:00.01	2	0	42	40	738K+312K	216K	216K
1490	alexander	0.0	00:00.00	4	0	14	14	148K+148K	148K	148K
1494	Google Chrome	0.0	00:45.07	1	93	97	48K	89K	89K	89K
1532	geditComp_138	0.0	00:00.00	1	0	22	22	224K+148K	148K	148K
1584	background_138	0.0	00:01.07	0	0	128	220	148K+148K	148K	148K
1585	background_138	0.0	00:00.00	0	0	128	220	148K+148K	148K	148K
1586	Google Chrome	0.0	00:10.19	1	93	346	19K	87K	87K	87K
1587	Nautilus	0.0	00:00.00	1	0	14	23	104K+244K	436K	436K

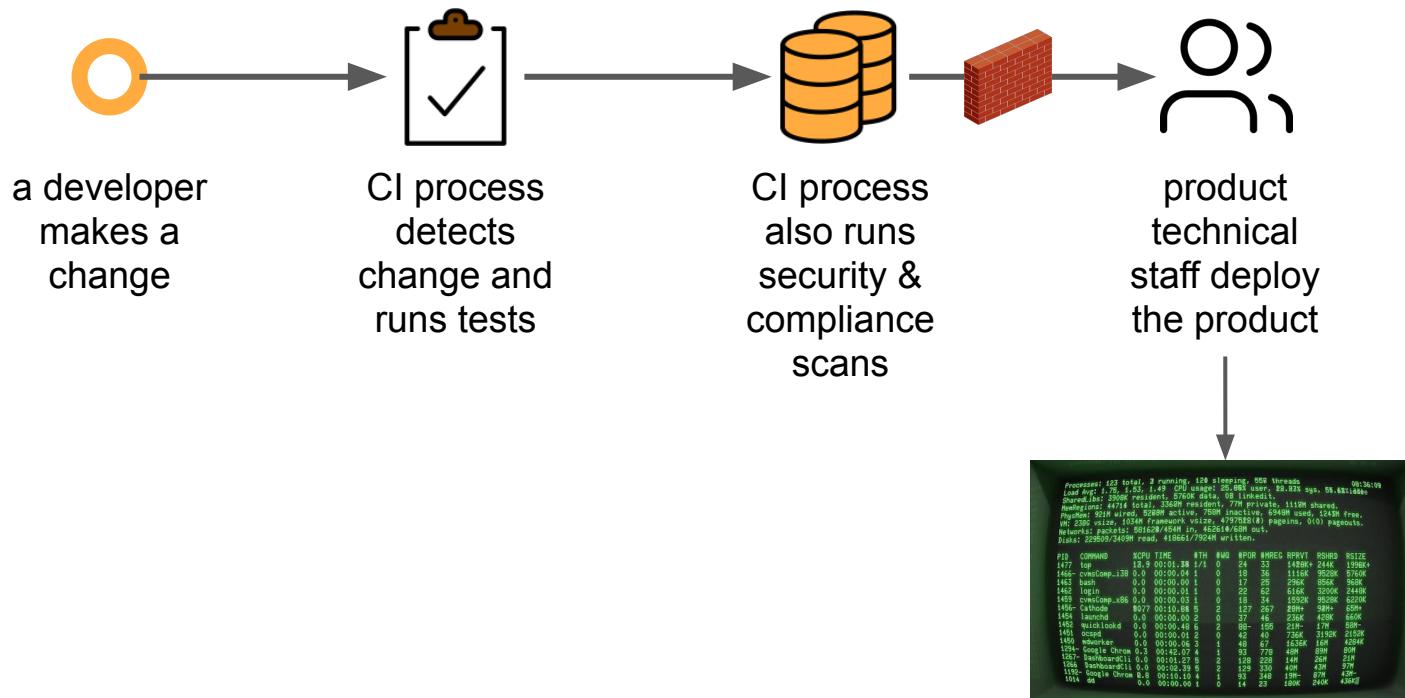
A continuous deployment process



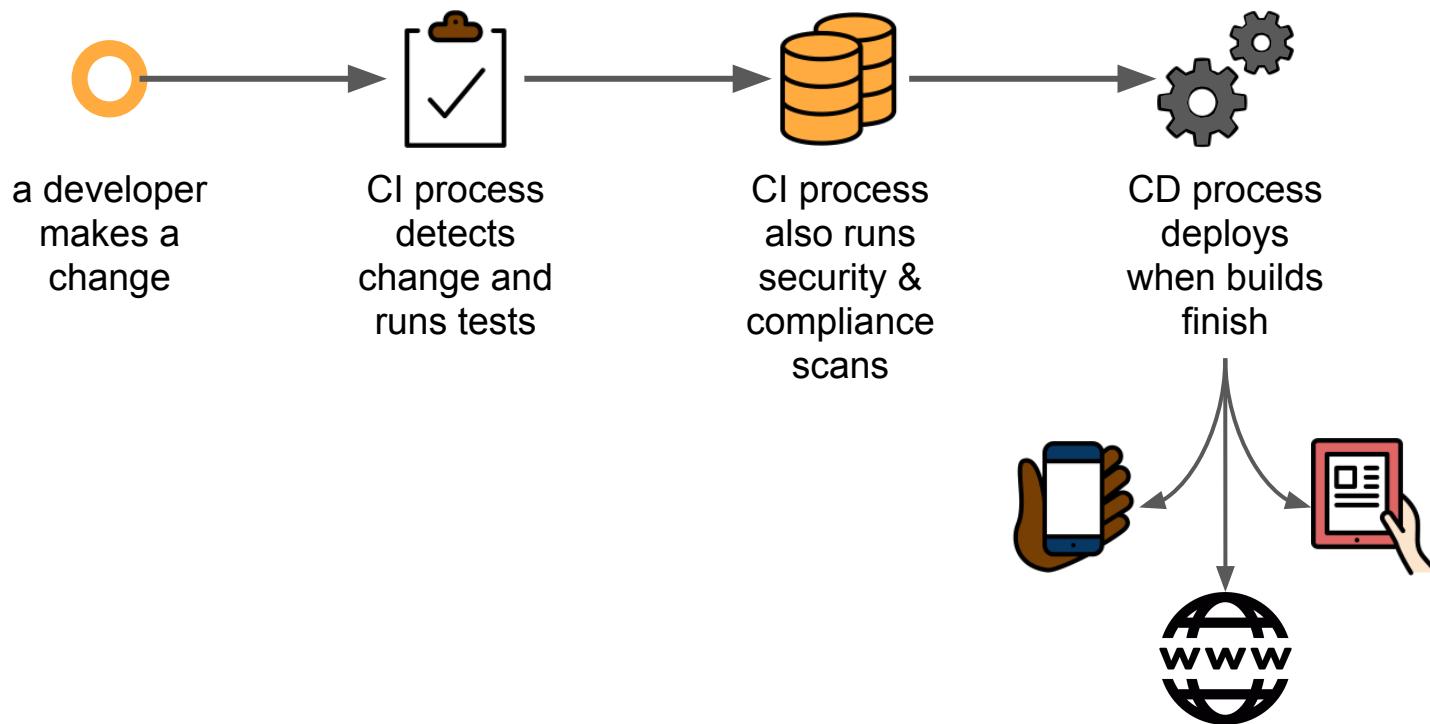
```
Processes: 123 total, 2 running, 120 sleeping, 824 threads
Load average: 0.00 0.00 0.00 CPU usage: 1.0% user, 0.0% nice, 0.0% sys, 55.0% idle
SharedLibs: 3908K resident, 5760K data, 68 K linkedin.
NativeLibs: 111M resident, 111M shared.
Memory: 3.1G wired, 3288M active, 765M inactive, 5349M used, 1024M free.
VM: 2305 voice, 1054M fram, 1054M voice, 2305M pages, 0(0) pagesout.
Swap: 0K used, 0K free, 0K available, 0K max.
DiskIO: 229509/3409M read, 418661/7924M written.
```

PID	COMMAND	XCPU	TIME	#TH	WNO	RPRIO	NREG	RPRIV	RSIGD	RSIZE
1477	top	13.9	00:01.38	1/1	0	24	33	148K+244K	159K+	159K
1445	gedomp_38	0.0	00:00.00	1	0	14	14	14K	14K	14K
1463	bash	0.0	00:00.00	0	0	17	25	256K	95K	95K
1462	login	0.0	00:00.00	0	0	23	62	616K	320K	244K
1459	gedomp_38	0.0	00:00.00	1	0	15	15	15K	15K	15K
1460	Cathode	0.0	00:01.08	5	2	127	267	284+	984+	65K+
1461	gnome-terminal	0.0	00:00.00	0	0	37	46	238K	42K	42K
1462	gnuchess	0.0	00:00.00	2	0	25	25	25K	25K	25K
1463	gnuchess	0.0	00:00.01	2	0	42	40	738K	312K	212K
1464	gnuchess	0.0	00:00.01	4	0	42	40	143K	143K	143K
1464	Google Chrome	0.0	00:45.07	1	0	93	97	48K	89K	89K
1465	Google Chrome	0.0	00:45.07	1	0	93	97	48K	89K	89K
1466	background2111	0.0	00:01.07	0	0	128	220	14K	42K	42K
1467	background2111	0.0	00:01.07	0	0	128	220	14K	42K	42K
1468	Google Chrome	0.0	00:10.19	1	0	93	346	19K	87K	87K
1469	Nautilus	0.0	00:00.00	1	0	14	23	104K	244K	430K

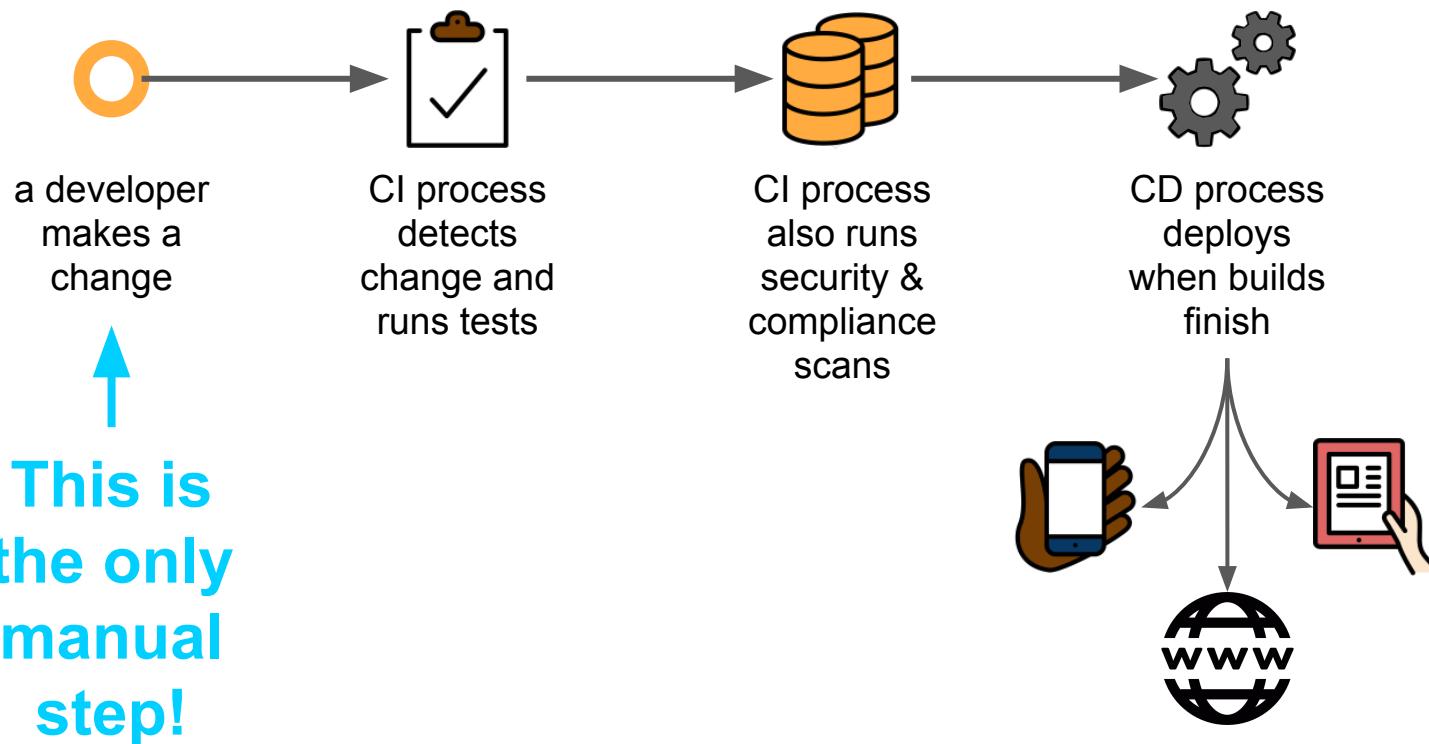
A continuous deployment process



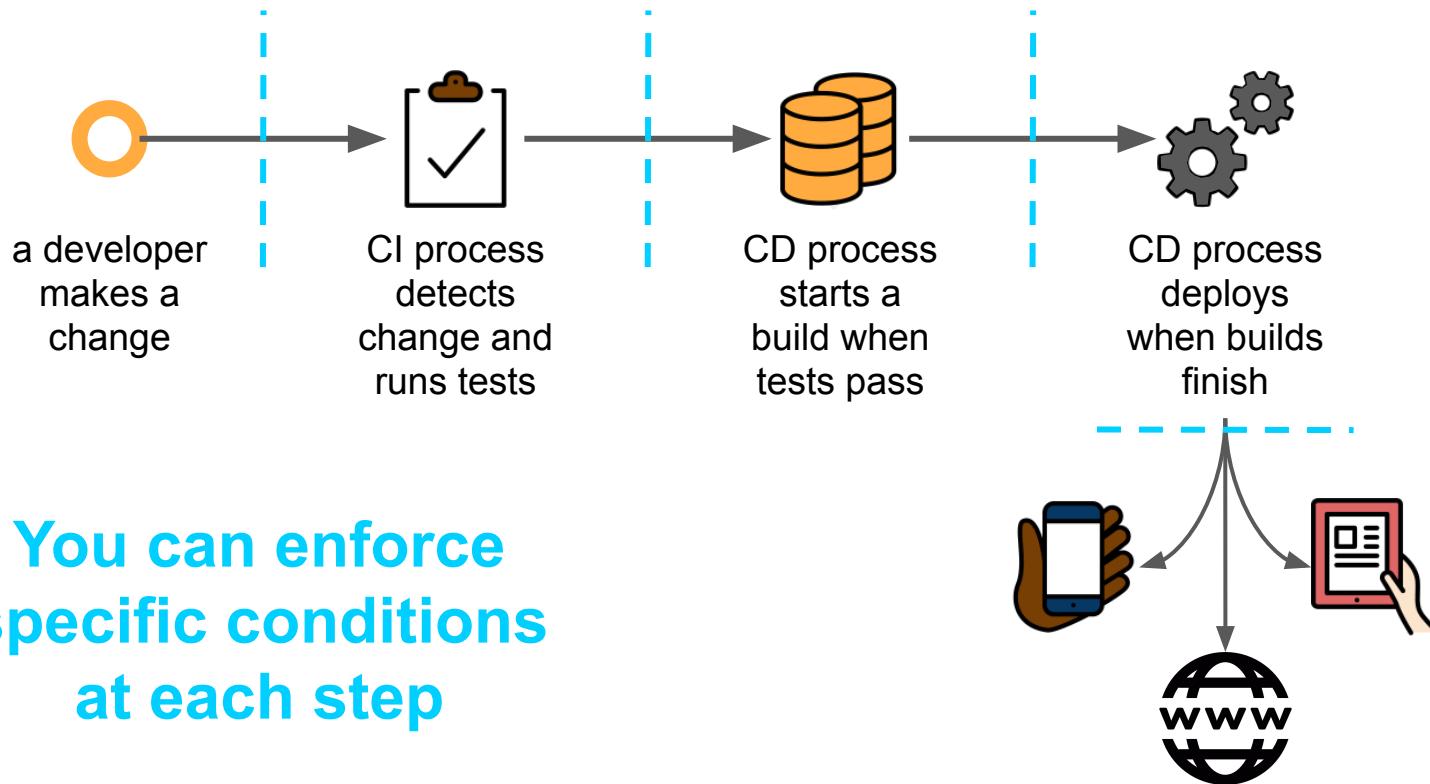
A continuous deployment process



A continuous, *automated* deployment process



A continuous, automated deployment process



The key takeaway:

**DevSecOps is an enabler for
continuous, iterative
improvement**

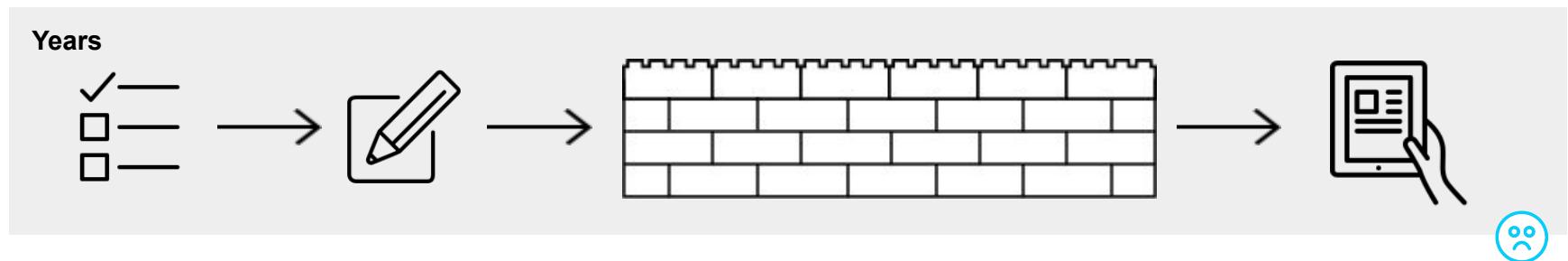
Agile acquisitions

Structuring RFPs to
support modern development
practices



The traditional way of approaching ‘big bang’ procurements

FROM



It's risky and makes juicy headlines

California junks \$179 million Medicaid IT modernization project with Xerox

New Hampshire's Medicaid Billing Computer System Still Having Glitches

The system has cost more than \$117 million since the first contract was approved in 2005 for about \$60 million, with the most recent amendment adding \$6.8 million to the price tag.

That's how it was supposed to work, anyway. What happened instead was a case of epic mismanagement that threatens to leave Texas taxpayers on the hook for more than \$130 million.

N.J. ends \$118M contract designed to ease enrollment in Medicaid and other welfare programs

R.I. Gov. Raimondo wants payment-for-performance contract with computer contractor

Oregon Health Authority director Lynne Saxton, whose agency was in charge of the \$166.7 million Medicaid enrollment system, defends the finished product.

Maryland fires firm upgrading Medicaid technology, may seek money back

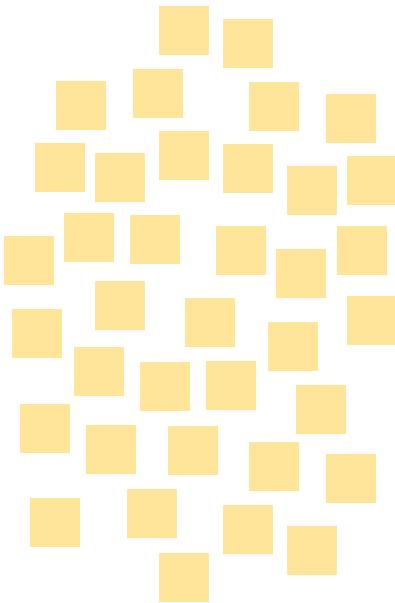
NEWS

Maine's Medicaid Mistakes

Audit: Xerox's Montana Medicaid project could be 6 years late

Thinking differently about requirements

User needs



Backlog



Delivery

Release 1



Release 2



Release 3



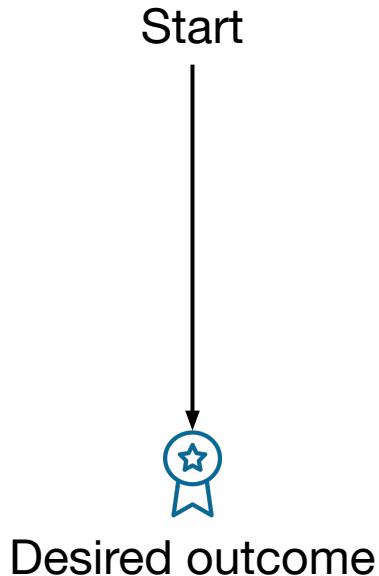
Release 4



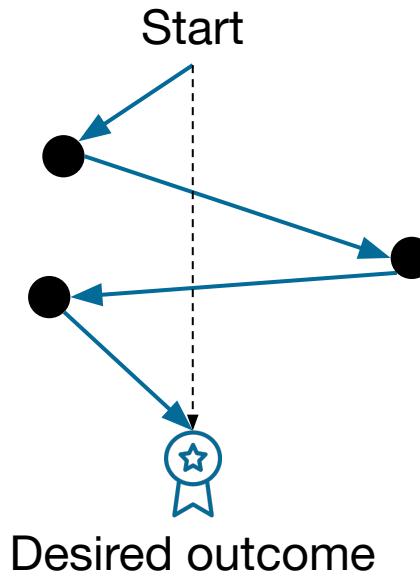
We know these elements need to be part of an agile software process.

Achieving a vision

Plan-driven

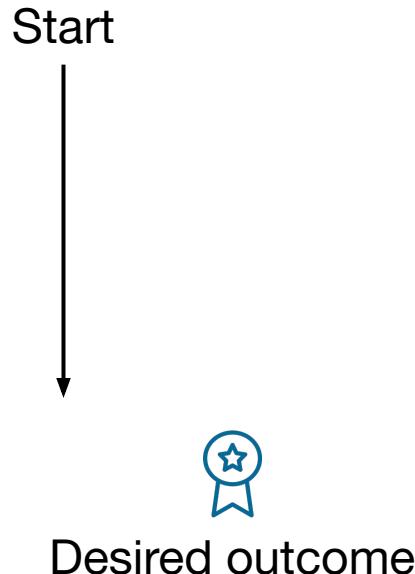


Iterative

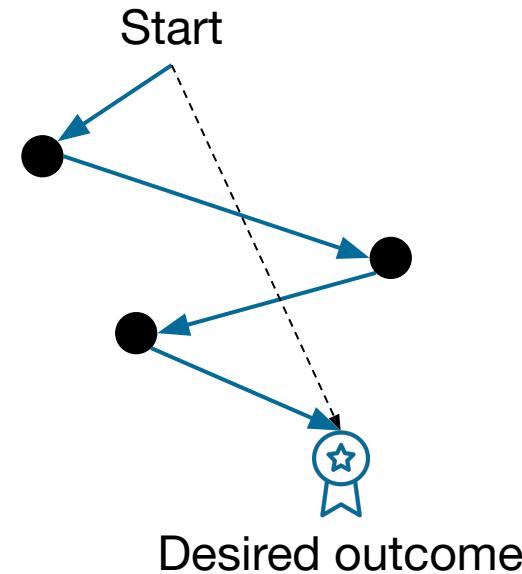


Achieving a vision

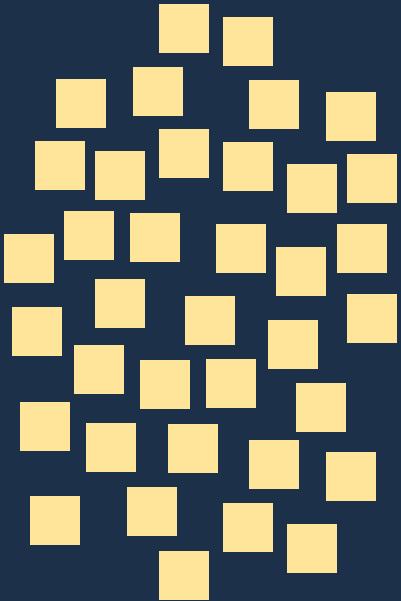
Plan-driven



Iterative



User needs



Backlog

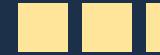


Delivery

Release 1



Release 2



Release 3

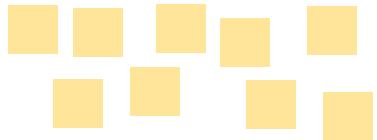


Release 4



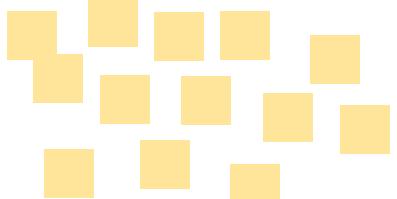
*Don't confuse the **destination**
for the **journey**.*

Organizational goals



+

User needs



Backlog



This is the
journey.
You pursue a
vision with a
dedicated
product team.

If you try to skip to
the end, **you've**
destroyed
iterative
modularity.

**Defining this up
front is waterfall.**





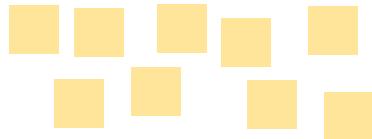
The traditional approach to solving the incompleteness problem is trying to do a “better” up-front inventory:

- More time spent
- More people involved
- More “requirements,” either actual or potential, captured in writing.

It's a fantasy that doesn't work.

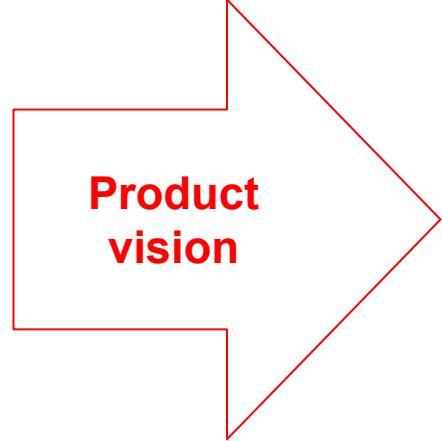
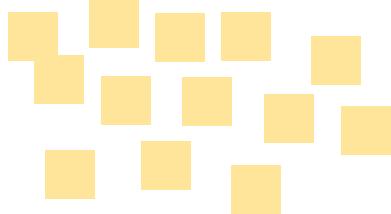
You cannot achieve a
different output unless
you change the input.

Organizational goals



+

User needs



Backlog



Delivery

Release 1



Release 2



Release 3



Release 4



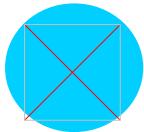
Two key artifacts bridge contracting and agile software development: the **product vision** and a **prioritized backlog** of work to do.

Contracts must also change to support this new, Agile approach



Adhering to Agile principles means that government as a **buyer must be descriptive, not prescriptive**.

Procurement Principles



Structure contracts to focus on **working software**, not planning documentation



Find vendors that use **human-centered design** to focus on software that meets people's needs



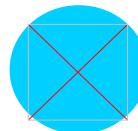
Leverage **open-source software**, and select modern tech stacks to ensure long-term sustainability



Structure contracts to **lower costs and avoid vendor lock-in**. Reduce switching costs.



Ensure **full transparency on the vendor team** - working code to actual users seen early and often



Collaboration with vendor teams vs. **constant negotiation**.

Remember what you're buying

- With agile contracting, you'll be buying teams of developers and other experts from vendors.
- Remember, this isn't a product — it's the skills and expertise required to make and maintain a product. That's what lets you be flexible.

What does quality look like?

Communicate with vendors about what you expect in working software.
For example, a Quality Assurance Plan should include measures:

- Tested Code
- Properly Styled Code
- Accessible
- Deployed
- Documented
- Available
- Secure
- User Research and Design Artifacts

How does this work in practice?

- **Outreach:** Find quality companies that you'd like to bid. Get them excited about what you're trying to do.
- **Transparency:** Get vendors' feedback on a draft RFP.
- **Brevity:** Don't ask for 100's of pages of proposals. Consider using verbal interviews to get a sense of team dynamic.
- **Post-award management:** Gov't owns the product decisions, but works collaboratively with the vendor. Use a Quality Assurance Surveillance Plan so vendors know what's expected from their research and coding activities.

**Questions
& discussion**