Agile Project Management
(or Burning Your Gantt Charts)

ESC-267 Michael Karlesky & Mark VanderVoord

The Zen Of “Done”

3 Things

1. Define Done

2. Feature Done

3. Project Done

Define “Done”

Done | dən |

adjective.

1. The code has been written according to the developers' interpretation of the spec (delivered 9 months ago). It can now be tested.

Done | **dən** |

adjective.

1. The code has been written but may require an unspecified amount of “cleanup” before shipping.

Done | dən |

adjective.

1. It works on a developer's machine.

Done | **dən** |

adjective.

1. Gantt chart progress has reached 100%
2. (more often) Gantt chart progress has reached 95% introducing a secondary state: “almost done”

Done | **dən** |

adjective.

1. The developers have stepped through the code with the debugger.

Done | dən |

adjective.

1. The debug build runs.

Done | dən |

adjective.

1. The state immediately preceding “done done”

Done | **dən** |

adjective.

1. A Zen-like continuation where the mind is at peace and the soul is at one with code, coworker, contractor, and customer alike

Done | **dən** |

adjective.

1. A Zen-like continuation where the mind is at peace and the soul is at one with code, coworker, contractor, and customer alike

Done | dən |

adjective.

1. A Zen-like state wherein all customer-specified features have been implemented to glowing accolades

Done | dən |

adjective.

1. A Zen-like state wherein all customer-specified features have been implemented to glowing accolades

Done | **dən** |

adjective.

1. A Zen-like state wherein all customer-specified features pass a vast (comprehensive) suite of unit, system, and acceptance tests

Done | **dən** |

adjective.

1. A Zen-like state wherein all customer-specified features pass a vast (comprehensive) suite of unit, system, and acceptance tests

Done | dən |

adjective.

1. A Zen-like state wherein
   (a) all customer-specified features pass unit, system, and acceptance tests
   (b) ...

Done | dən |

adjective.

1. A Zen-like state wherein
   (a) ...
   (b) It works on more than just my machine

Done | dən |

adjective.

1. A Zen-like state wherein
   (a) ...
   (b) It works on more than just my machine

Done | dən |

adjective.

1. A Zen-like state wherein
   (a) ...
   (b) builds are reproducible, identical, and machine independent

Done | dən |

adjective.

1. A Zen-like state wherein
   (a) all customer-specified features pass unit, system, and acceptance tests
   (b) builds are reproducible, identical, and machine independent


Define “Feature Done”

Feature?

Feature | `'fē ch ər` |

noun.

1. Something that a customer can actually define in a concrete way, crossing that mystical gap between geek and non-geek.

Feature | `'fē ch ər` |

noun.

1. Something that a customer can actually define in a concrete way, crossing that mystical gap between geek and non-geek.

Feature | `'fē ch ər` |

noun.

1. Something that a customer (a) can explain (b) can know when it has been done correctly and is properly Awesome

Feature | `'fē ch ər` |

noun.

1. Something that a customer (a) can explain (b) can know when it has been done correctly and is properly Awesome

Feature | **ˈfē ch ər** |

noun.

1. Something that a customer (a) can explain (b) can verify

Feature | **ˈfē ch ər** |

noun.

1. Something that a customer (a) can explain (b) can verify (c) feels is worth some kind of money or compensation to develop

Feature | **ˈfē ch ər** |

noun.

1. Something that a customer (a) can explain (b) can verify (c) feels is worth some kind of money or compensation to develop

Feature | **ˈfē ch ər** |

noun.

1. Something that a customer (a) can explain (b) can verify (c) will pay for

Customer?

Customer

noun. synonyms:

1. End User

Customer

noun. synonyms:

1. End User
2. Engineering Department

Customer

noun. synonyms:

1. End User
2. Engineering Department
3. Marketing

Customer

noun. synonyms:

1. End User
2. Engineering Department
3. Marketing
4. Sales

Customer

noun. synonyms:

1. End User
2. Engineering Department
3. Marketing
4. Sales
5. Production

Customer

noun. synonyms:

1. End User
2. Engineering Department
3. Marketing
4. Sales
5. Production
6. Technical Support

Since We Understand Feature

Now For A Round Of
“Is That A Feature?”

?

SPI Interface Driver

?

Application should calculate and display the air-speed velocity of an unladen swallow

?

Application Should Initialize ADC as Pulse Convert Mode

?

Application Should Filter Results Before Display

?

Application Should Look Awesome

?

Application Should Blink A Warning Light Before It Crashes

We Know Features

So When Is The Feature Done?

Done | **dən** |

adjective.

1. A Zen-like state wherein
   (a) all customer-specified features pass unit, system, and acceptance tests
   (b) builds are reproducible, identical, and machine independent

unit, system, acceptance

unit, system, acceptance

prove pieces of code work

unit, system, acceptance

prove the release build works

unit, system, acceptance

proves to customer

the features work

development team

now also develops tests

strangely

does NOT

take more time

does NOT

make the project more complex

plus

encourages good design

encourages modular code

and

(obviously?)

promotes TESTABLE code

when all tests

we know those features

DONE

Done For Good?

Yes!

Continuous Integration

Every Time

Add A Feature

Every Time

Fix A Bug

Every Time

We Change Anything

ALL TESTS

Re-run

Re-validated

Re-verified

But We Can't Work On Features Until There Is Infrastructure

Right?

Wrong

“Simplest Thing That Works”

Then

Refactor As Needed

You Can't Guess

Exactly

How All Modules Will Be Used

Quite Likely

“You Ain't Gonna Need It”

When You're Wrong

Spend More Time Changing

Than Writing

But You Skipped The Step Where We Wrote A System Design Specification!

Wrong

Develop Documentation

Like Software

Collaborative

Fast

Unobtrusive

Get the Customers Involved

Don't Let It Feel Rigid

Don't Let It Get Stale

Always Evolving

Accurate

wiki

That's Feature Done

Define “Project Done”

Project Done | **präj ˌekt dən** |
noun.

Project Done | **präj ˌekt dən** |
noun.

1. A Zen-like state wherein
   (a) all customer-specified features pass unit, system, and acceptance tests
   (b) builds are reproducible, identical, and machine independent

anti-climactic?

really want to know

WHEN

Project Done

Let's Start With

Estimation

All Features Given Points

Points are NOT Hours

Points are Complexity

If Feature 1

$2x$

Feature 2

Then

$2x$

Points

Humans Better At Estimating Complexity Than Time

Estimating Points

Factors Out

Overhead

Overhead

Meetings

Documenting

Overhead

Meetings

Documenting

Requirement Gathering

Overhead

Meetings

Documenting

Requirement Gathering

Overhead

Meetings

Coffee Breaks

Phone Calls

Documenting

Requirement Gathering

Overhead

Meetings

Coffee Breaks

Phone Calls

Documenting

Requirement Gathering

Overhead

Hear Ryan's Joke

Meetings

Coffee Breaks

Phone Calls

Documenting

e-mail

Requirement Gathering

Overhead

Hear Ryan's Joke

Meetings

Coffee Breaks

Phone Calls

Documenting

e-mail

Requirement Gathering

Overhead

Hear Ryan's Joke

Meetings

YouTube

Coffee Breaks

Phone Calls

Documenting

e-mail

Requirement Gathering

Overhead

Hear Ryan's Joke

Meetings

YouTube

Coffee Breaks

less-than-productive time

Phone Calls

Documenting

e-mail

Requirement Gathering

Because The Truth Is

Hear Ryan's Joke

Meetings

YouTube

Coffee Breaks

less-than-productive time

No One Is Productive 100% Of Their Time

Points

Estimate

Net Progress

So How Do We Assign Points?

Planning Poker

Step 1

As A Group

Choose a Medium-Size Feature

Call it 5

Everyone Has Deck of Cards

Numbered 1-10

(Or 1-2-3-5-8-13)

For Each Feature

Quickly Discuss It

Then, Everyone Chooses a Card

(But Does Not Show It)

Everyone Reveals

If Points are Close

(within a card)

Use the Highest Value

Otherwise

Re-Discuss

(there's a difference in view)

(<u>what</u> the feature is)

(the risks involved)

(or what's unknown)

After Discussion

Re-Select Cards

Repeat Until You Agree

Repeat Until All Features Estimated

That's It?

What about Features we don't understand yet, or high risk Features?

Good Point

Higher Risk Features

Get Higher Points

Lack of Definition

Gets Higher Points

Some of Those Unkowns

Will be Easy

Some will be Scary

In Fact,

Normal Features

Will be Easier / Harder

Than Estimated

But That's OK

(we'll see why in a moment)

Now For A Round Of
“How Many Points?”

Our Features:

1. Return Version Message via serial port when requested.
2. Measure Filtered Battery Voltage and Return 1/sec
3. Estimate Battery Life Remaining and Return 1/sec
4. Blink Red LED when Battery Is Too Low
5. Send Out LOW Message once when Battery Too Low

So Now What?

It's Time For Fancy Math

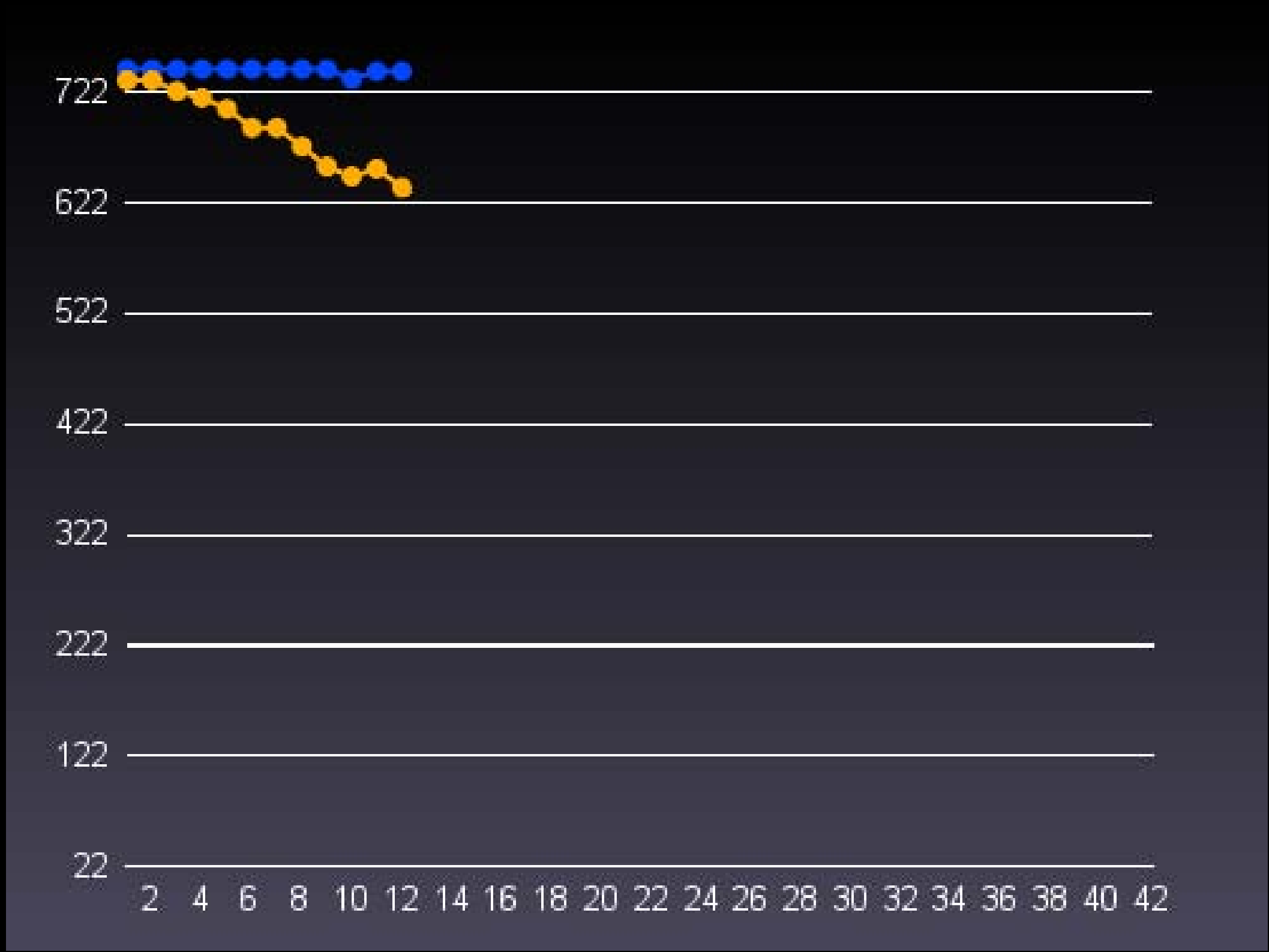
We Sum The Points

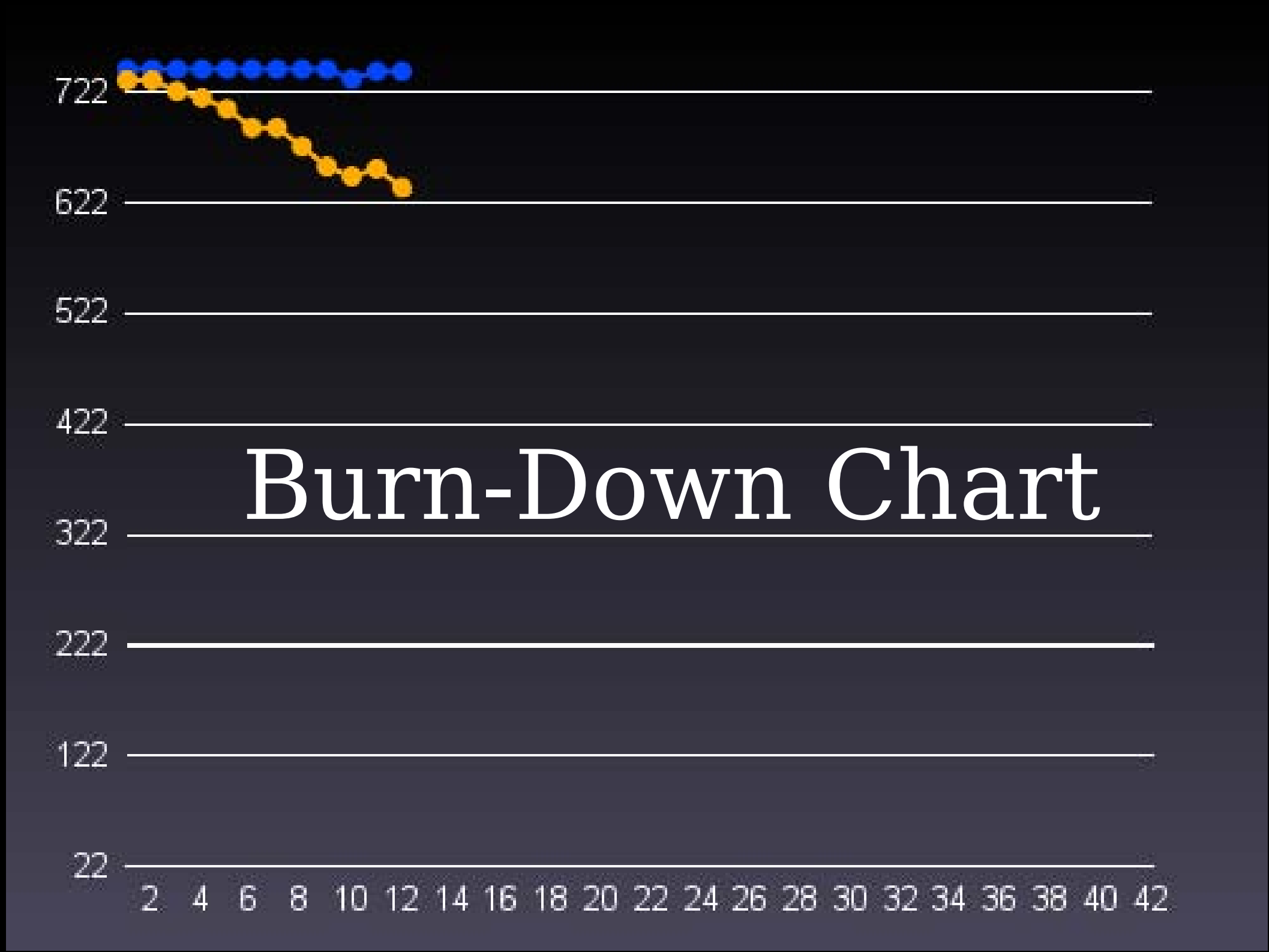
As We Work

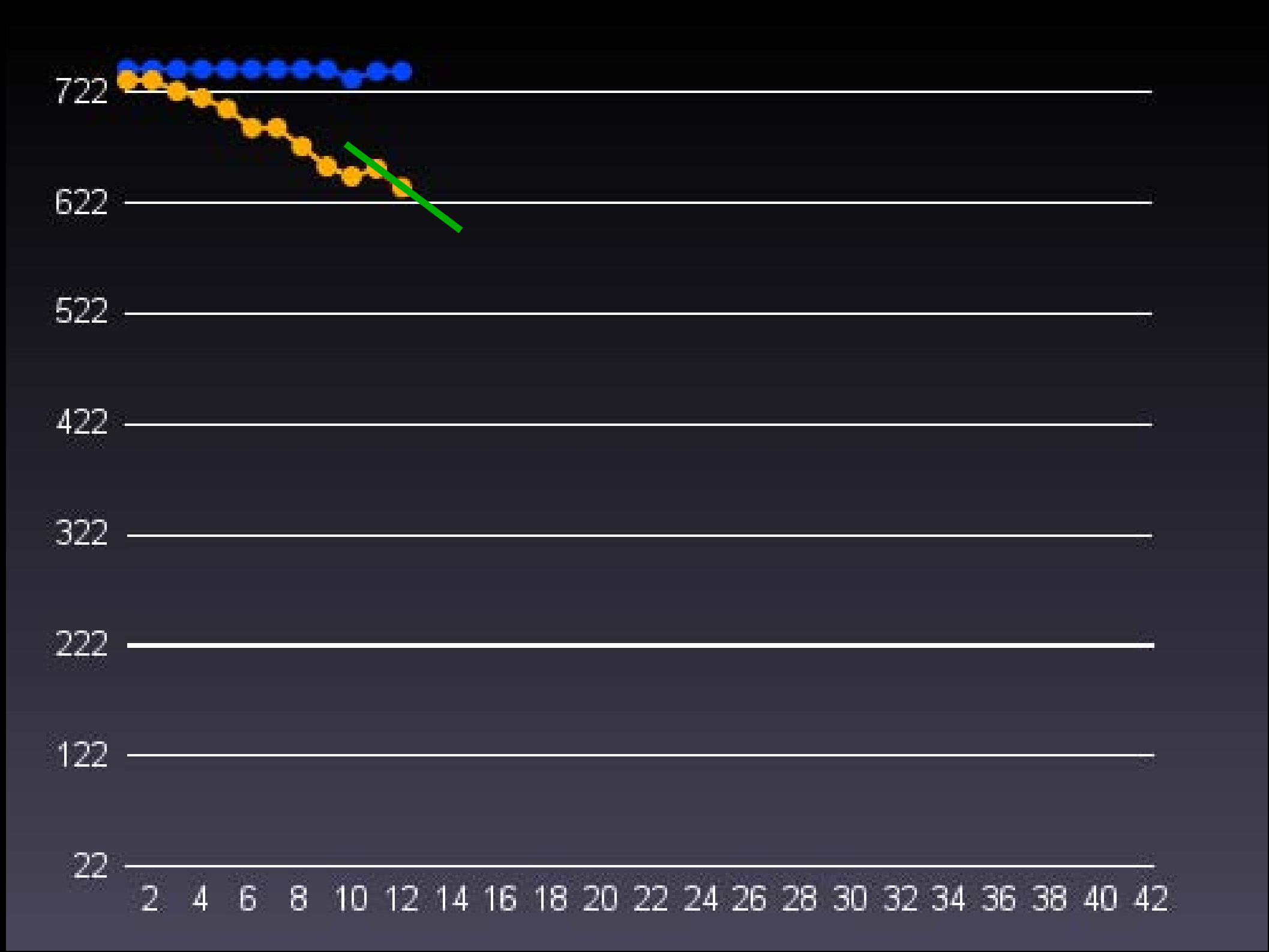
Track Points From “Features Done”

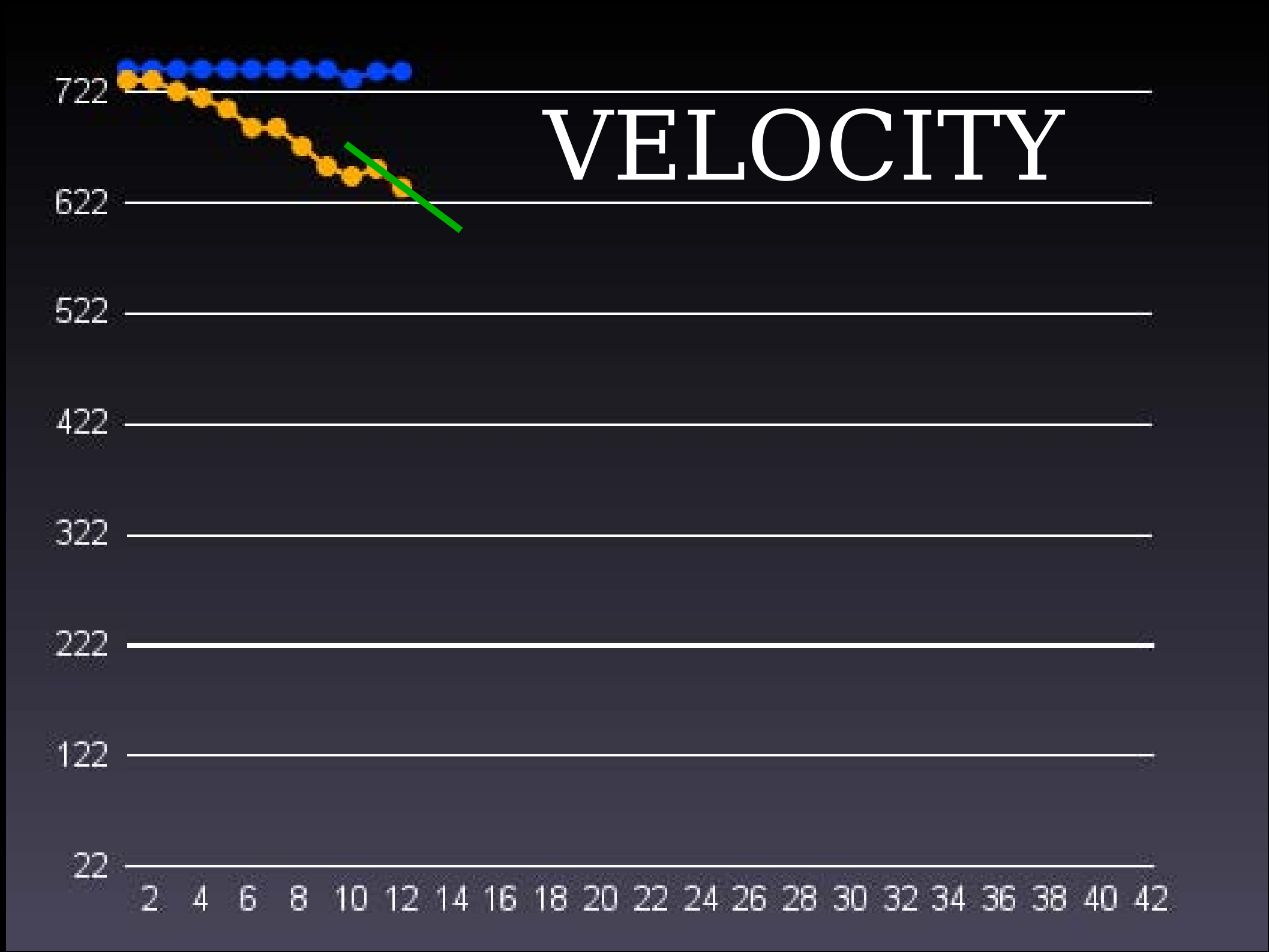
At Regular Intervals

(Iterations)


722
622
522
422
322
222
122
22
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42


Burn-Down Chart
722
622
522
422
322
222
122
22
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42


722
622
522
422
322
222
122
22
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42


VELOCITY
722
622
522
422
322
222
122
22
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42

Variations in Points Completed

Per Iteration

Average Out

We Use Points Completed

In Last Iteration

(or Average of Last Few)

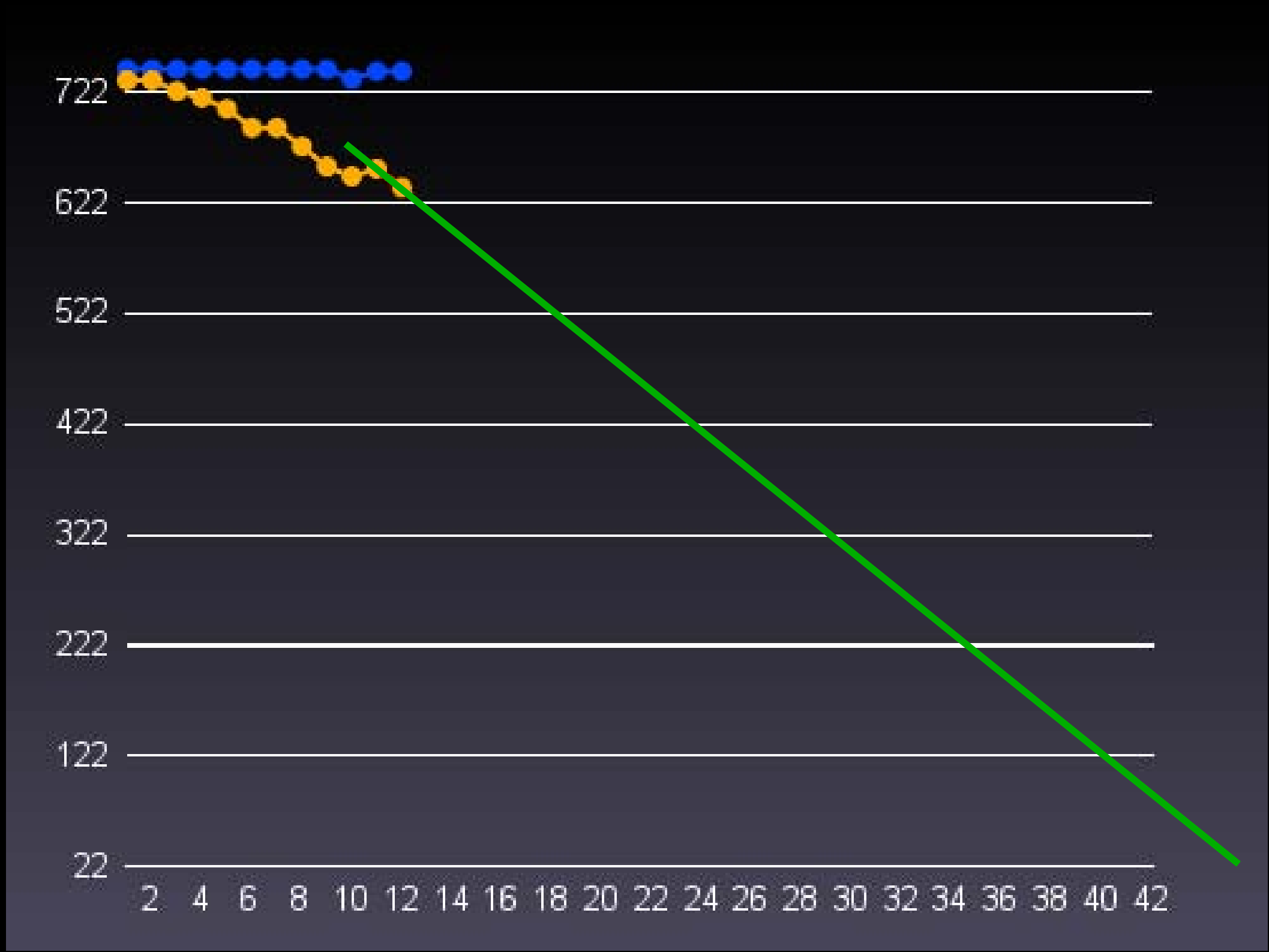
Calculate Velocity

(Points / Iteration)

Use Velocity

From Points Remaining

To Predict “Project Done”


722
622
522
422
322
222
122
22
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42

I'm Sorry

That End Date Is Not Acceptable.

We Have Options

We're Tracking Velocity

We Know Effect Of Adding People

We Know Effect Of Dropping or Modifying Features

We Can Estimate Effect Of Buy/Build Decisions

Huh?

Feature Sets Change

New Features Are Requested

Features Were Forgotten During Estimation

Priorities Change

Each Time

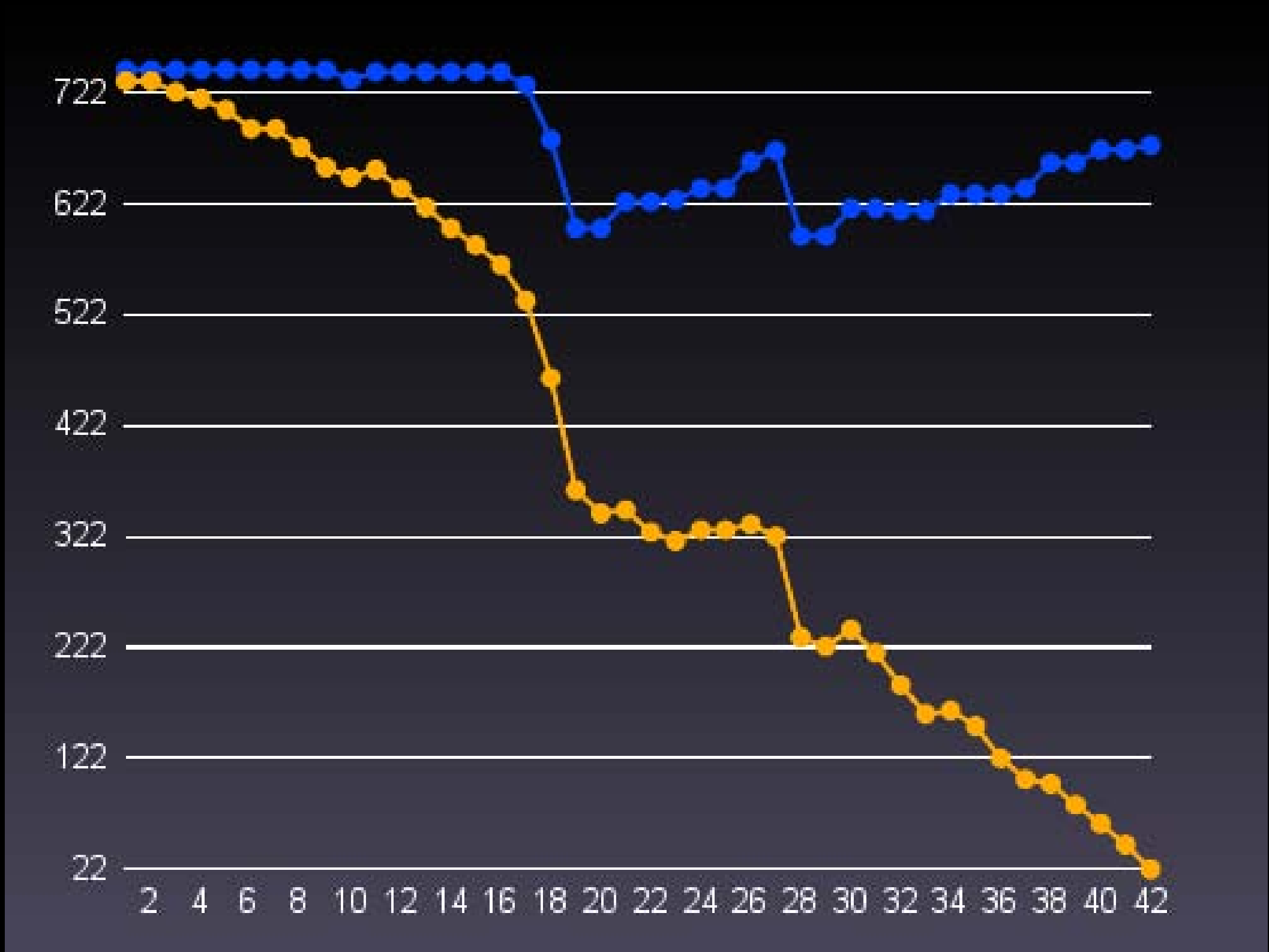
We Add / Remove

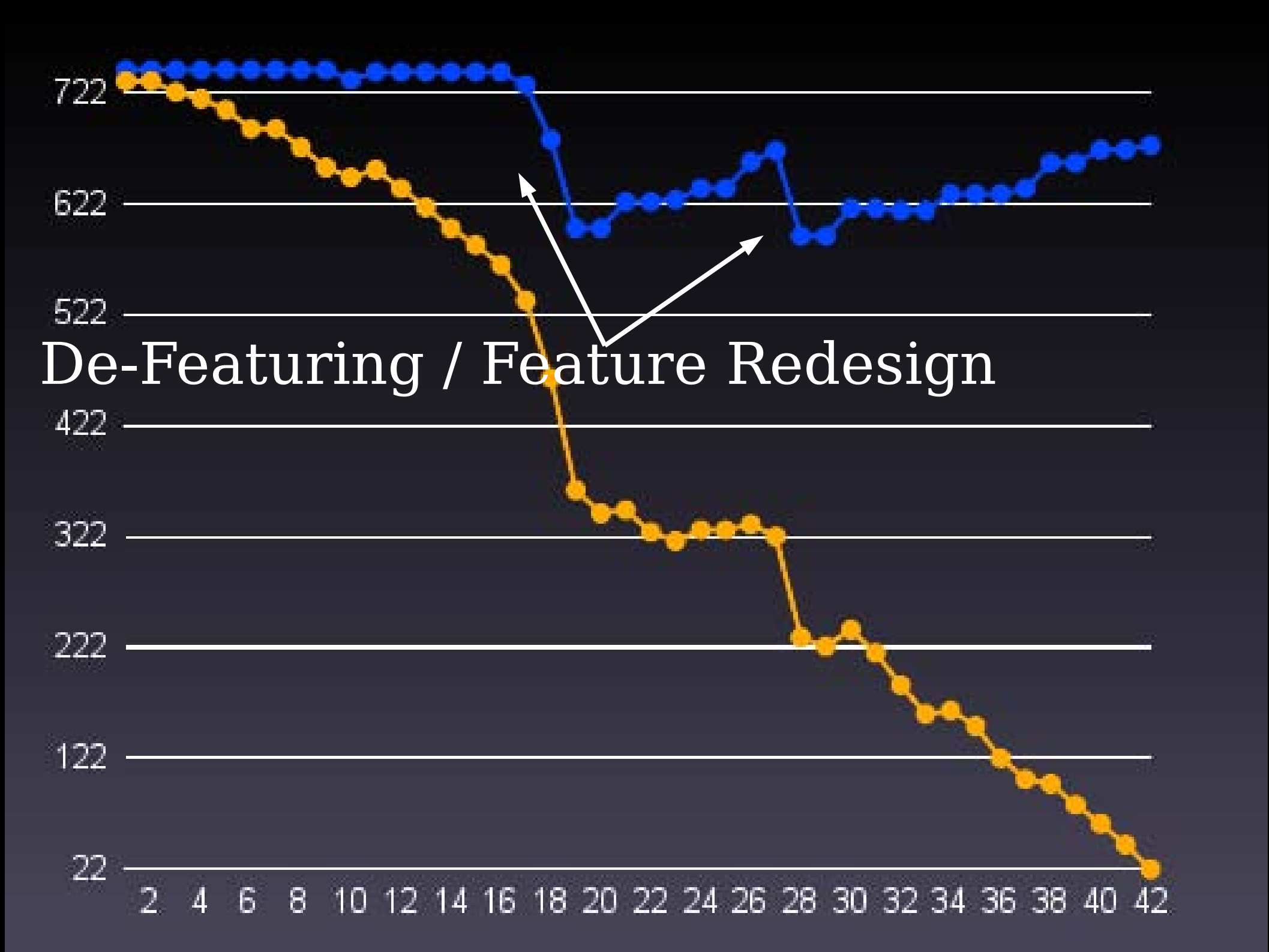
Points to Total Points

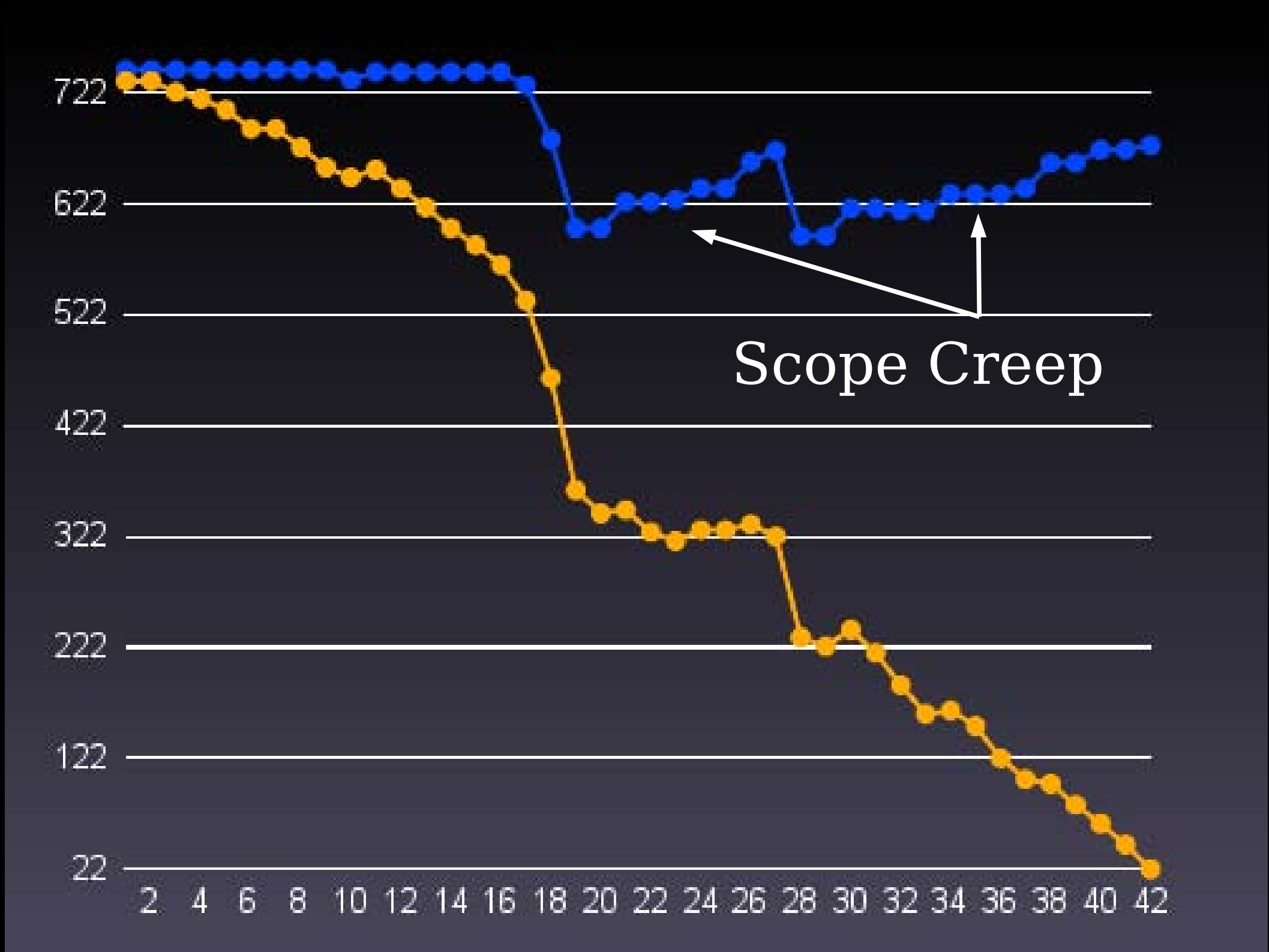
Points Remaining

Giving Us Updated Estimates

Automatic History

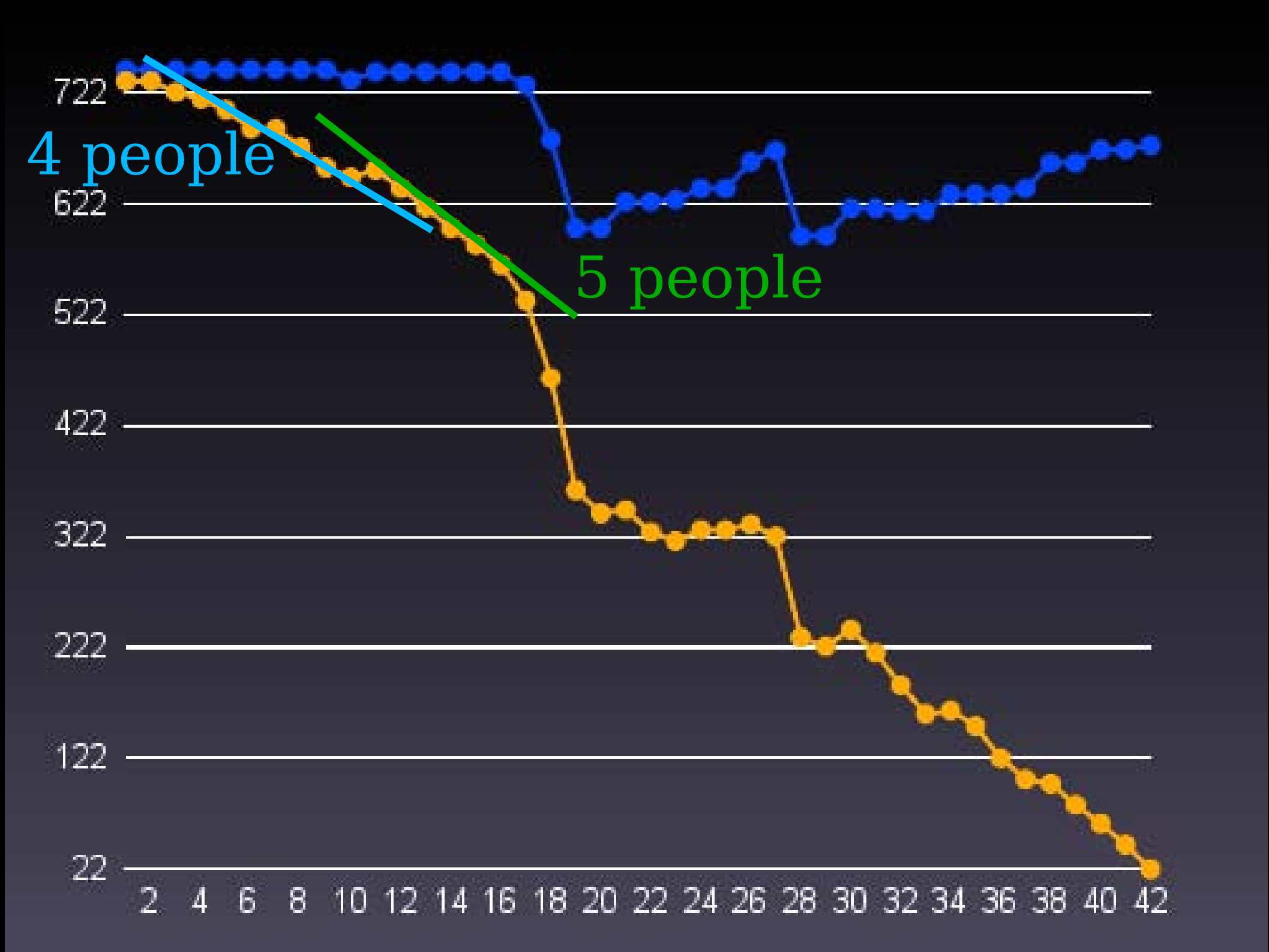

722
622
522
422
322
222
122
22
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42


722
622
522
422
322
222
122
22
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42
De-Featuring / Feature Redesign

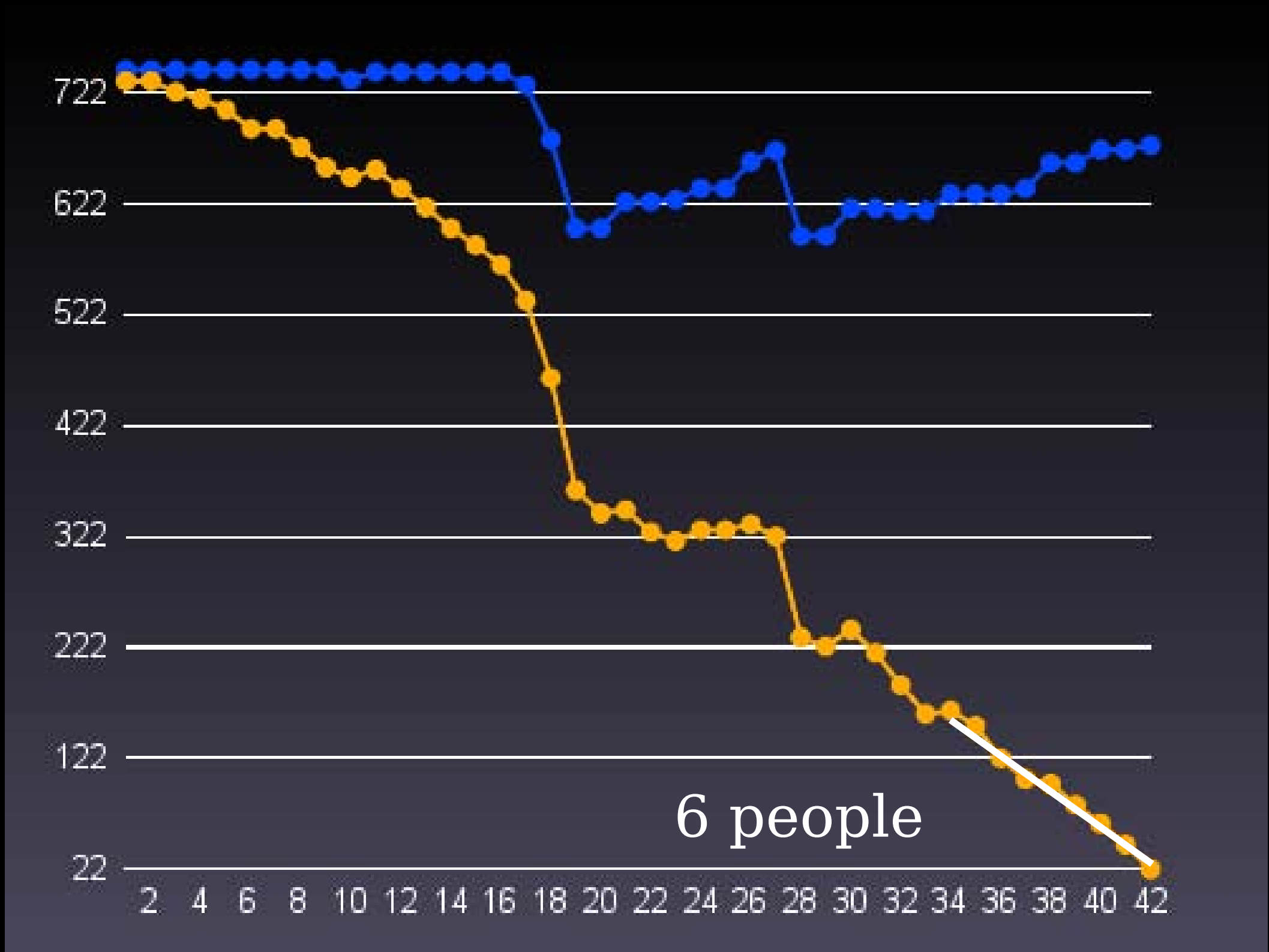

722
622
522
422
322
222
122
22
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42
Scope Creep

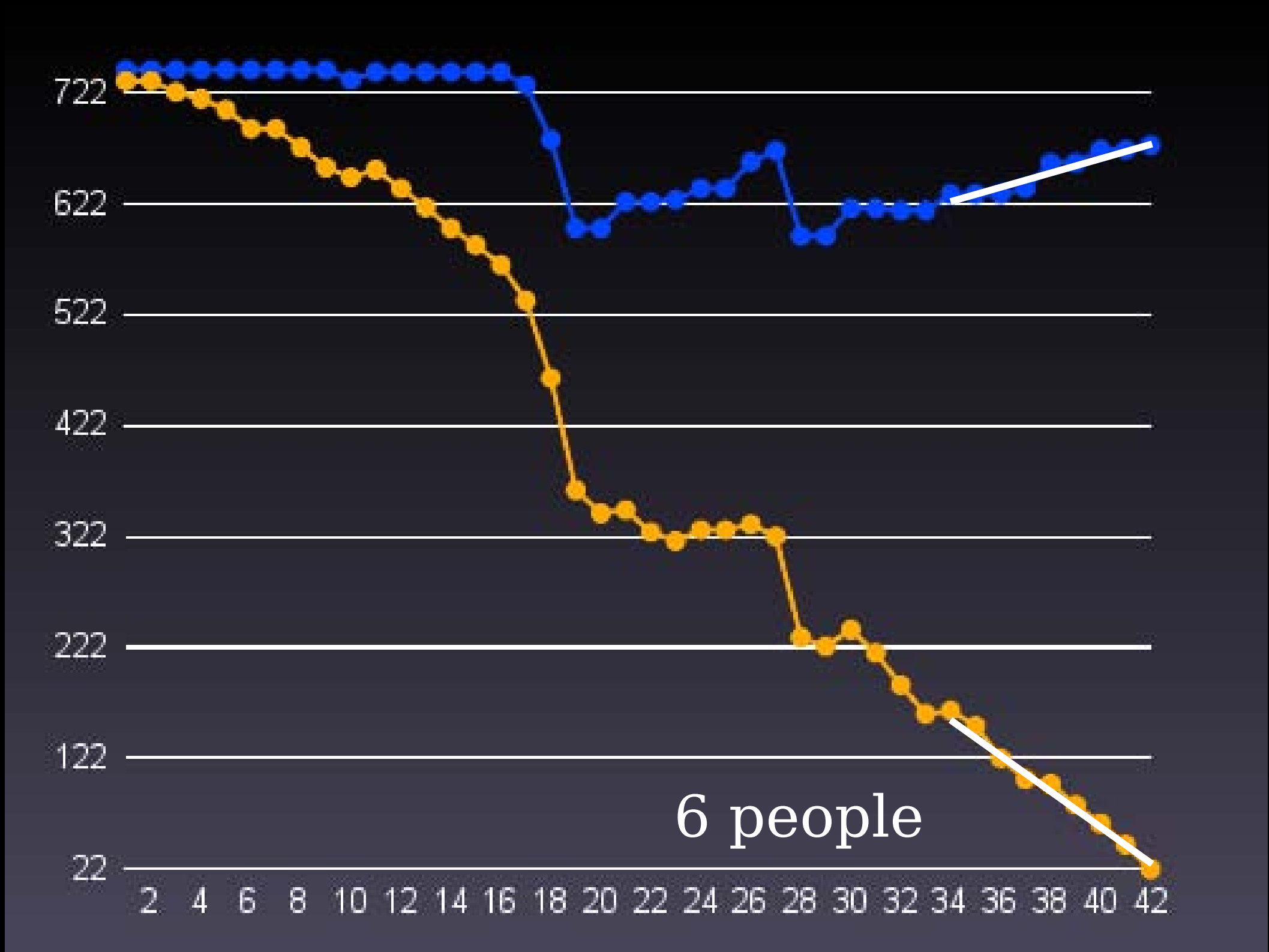

722
622
522
422
322
222
122
22
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42
"Project Done"

What About Adding People?


4 people
5 people
722
622
522
422
322
222
122
22
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42

Be Careful


722
622
522
422
322
222
122
22
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42
6 people


722
622
522
422
322
222
122
22
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42
6 people

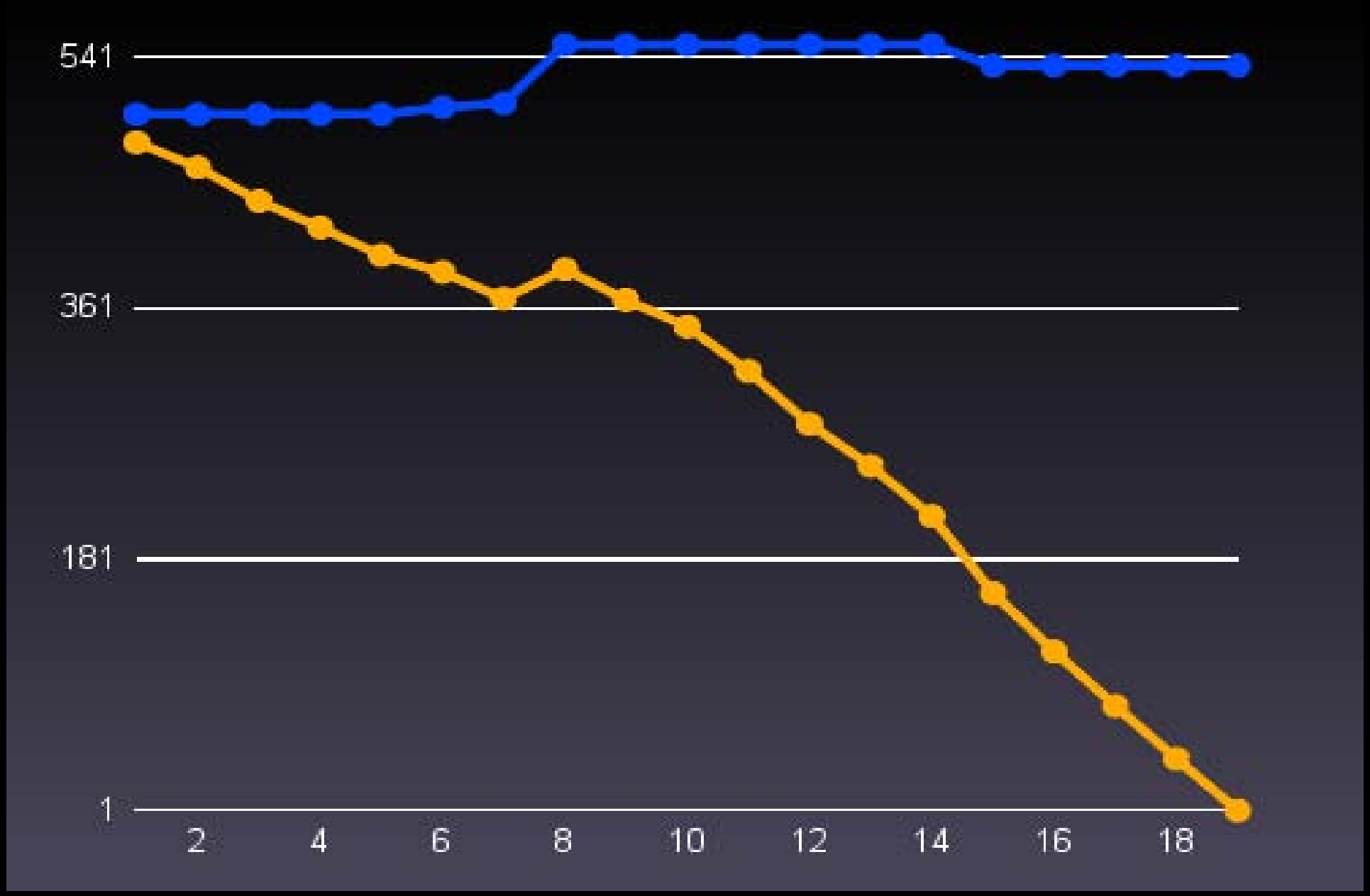
Because of This

Often Add Another Graph

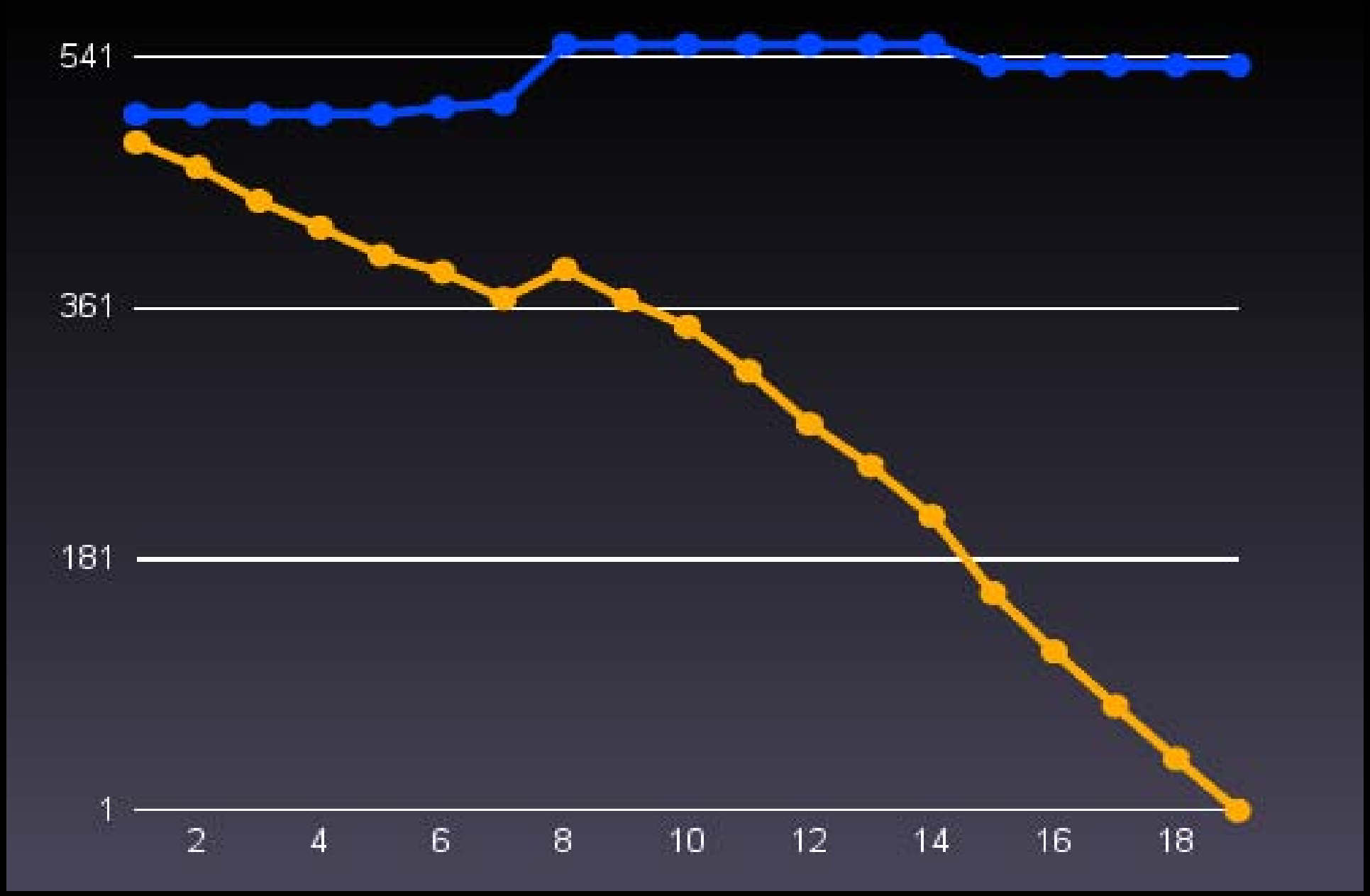
Track Velocity

Now For A Round Of
“What Happened?”

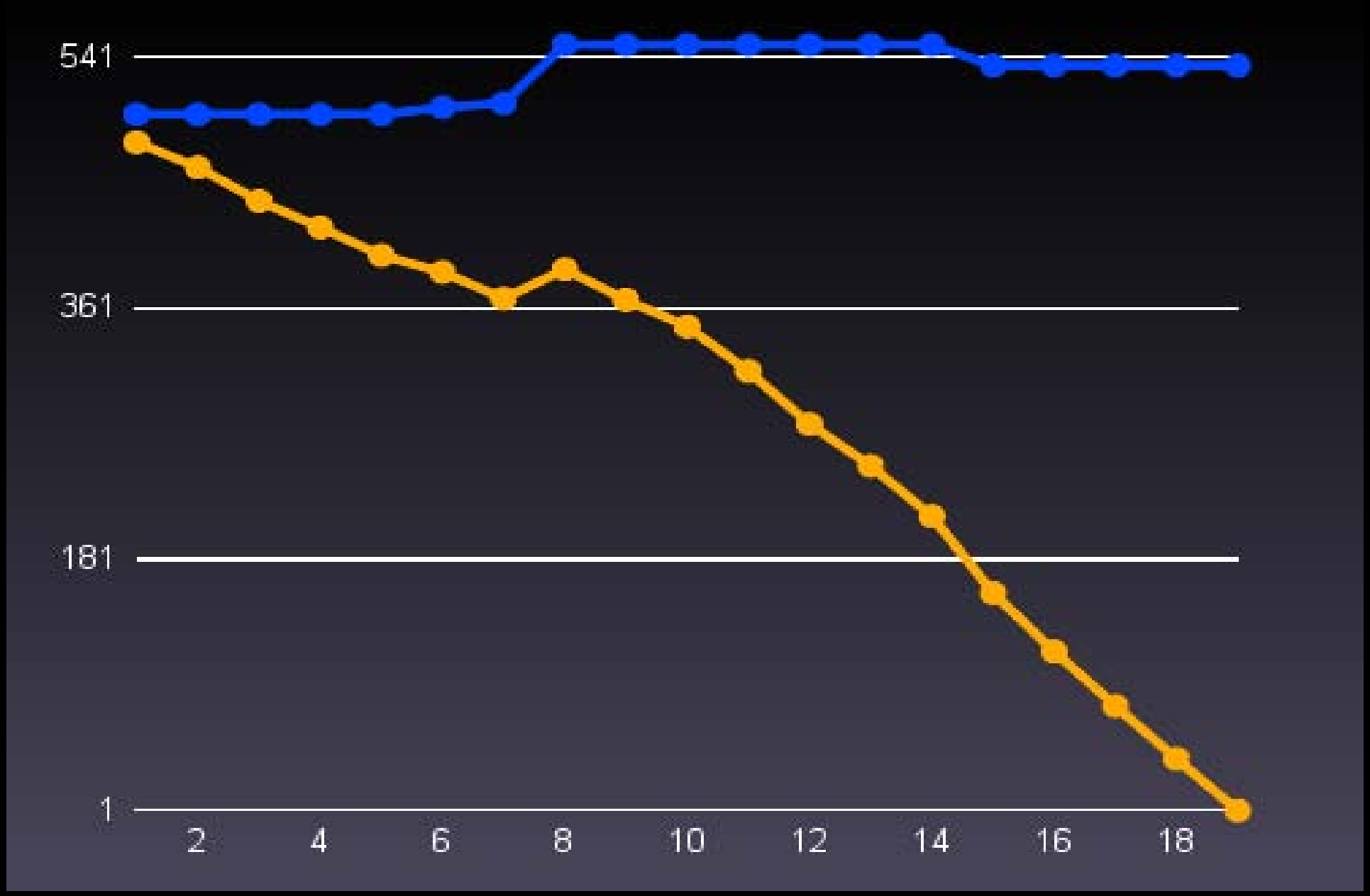
Where Did We Add Features?



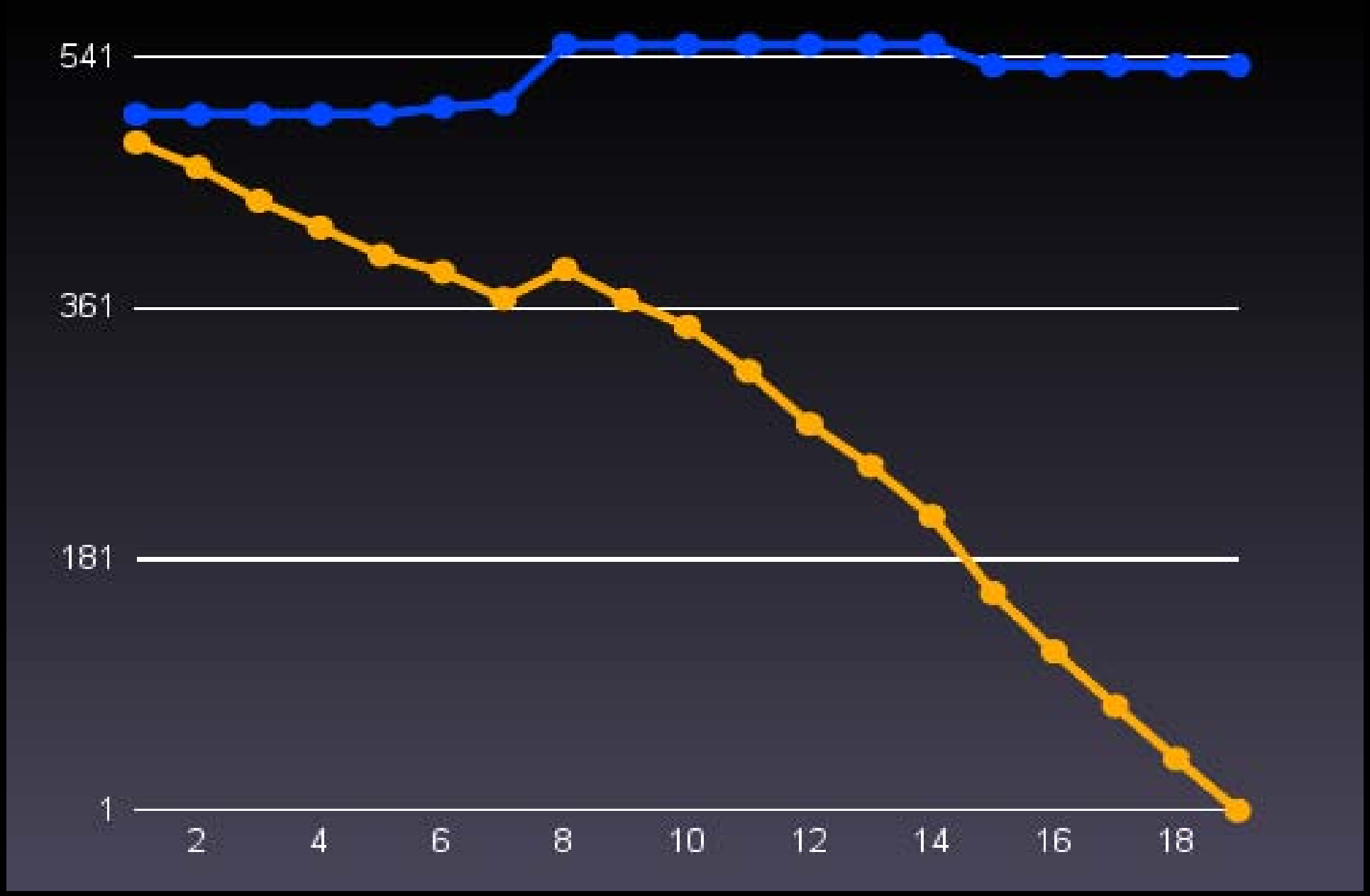
What Was Our Initial Estimate?



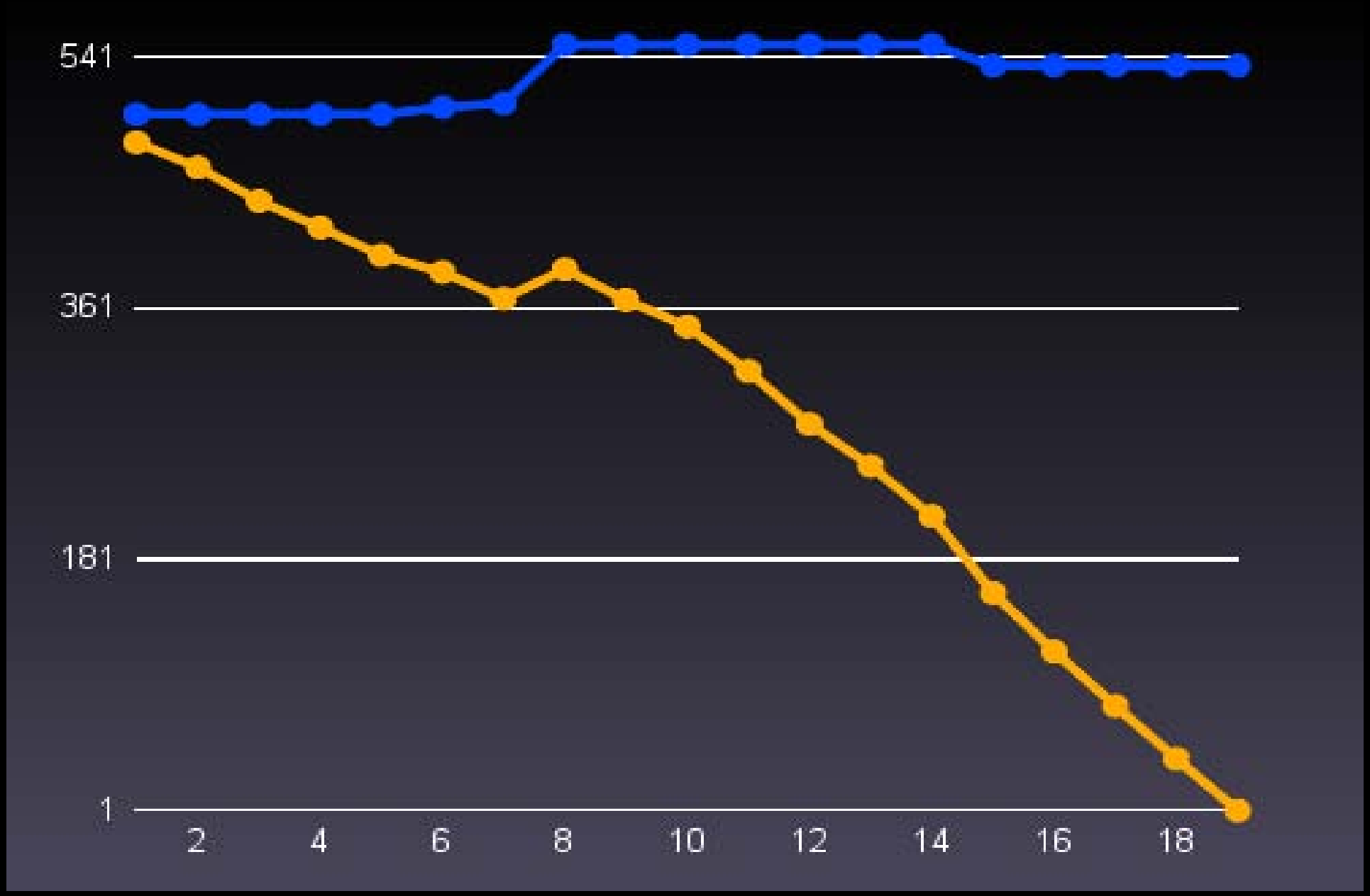
What Was Velocity At Iteration 4?



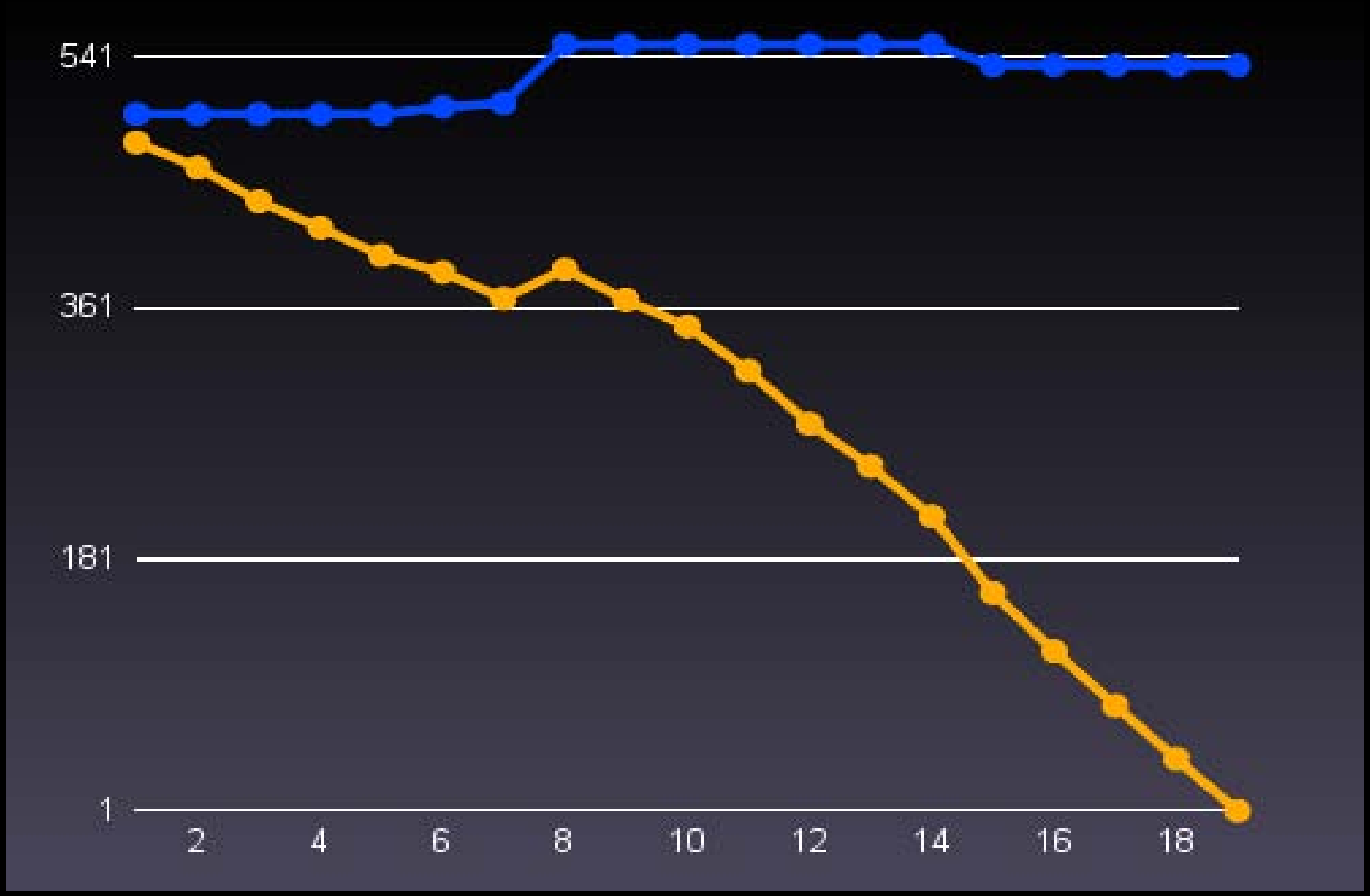
Where Did We De-Feature?



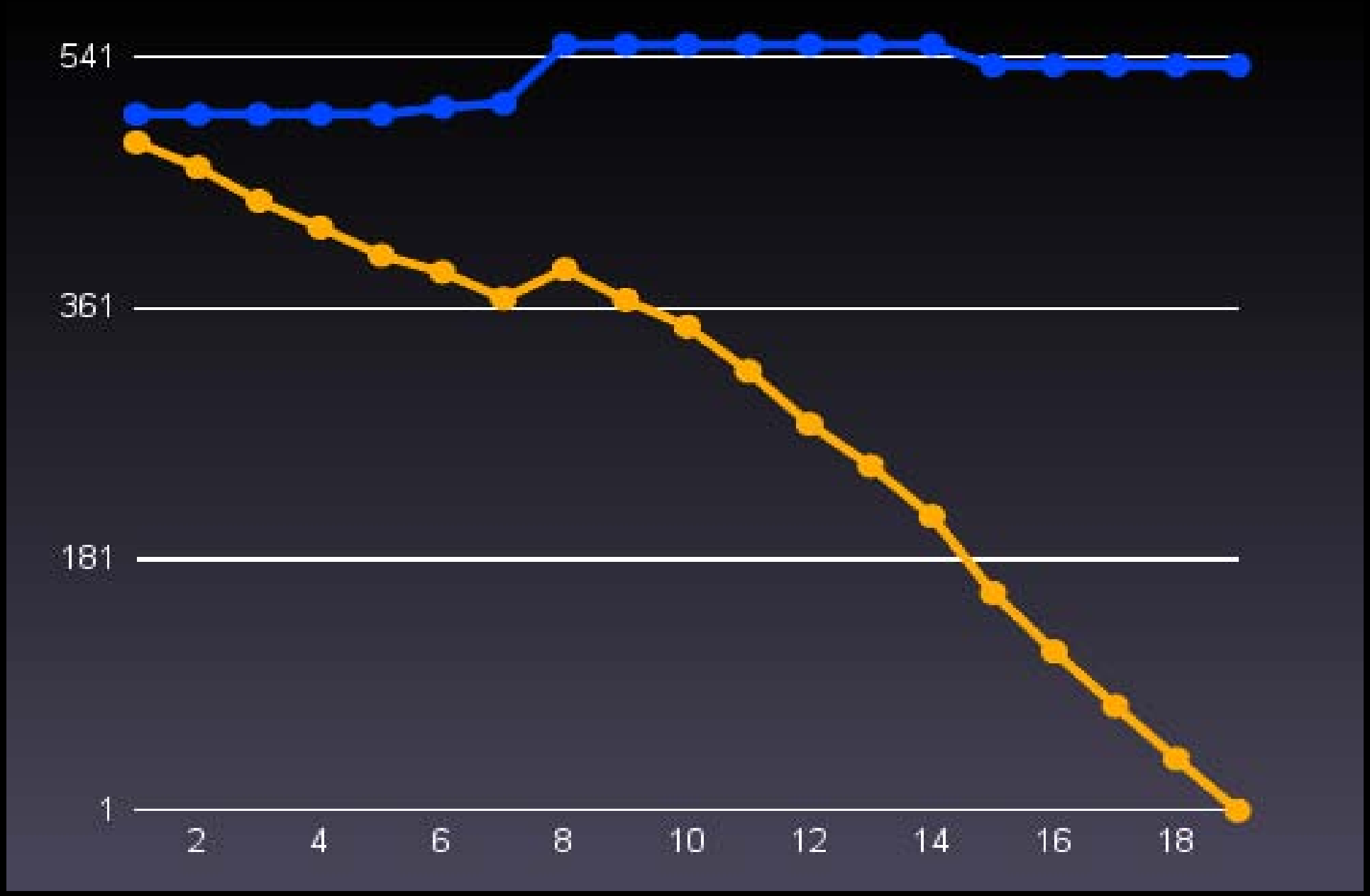
What Was Our Final Total?



Where Was A Velocity Jump?



Where Were We Done?



So We Just Work Through Points Until We're Done?

Almost

The Order Is Important

How Do We Set Priority?

Every Iteration

Work With Customer

Prioritize

Most Important Features First

Highest Risk First

Greatest Unknowns First

(Helps Flush Out Definition)

Plus a Bonus

Stop When You Want

If Customer Happy

Current Feature Set

All Tests Pass

Then Ship It

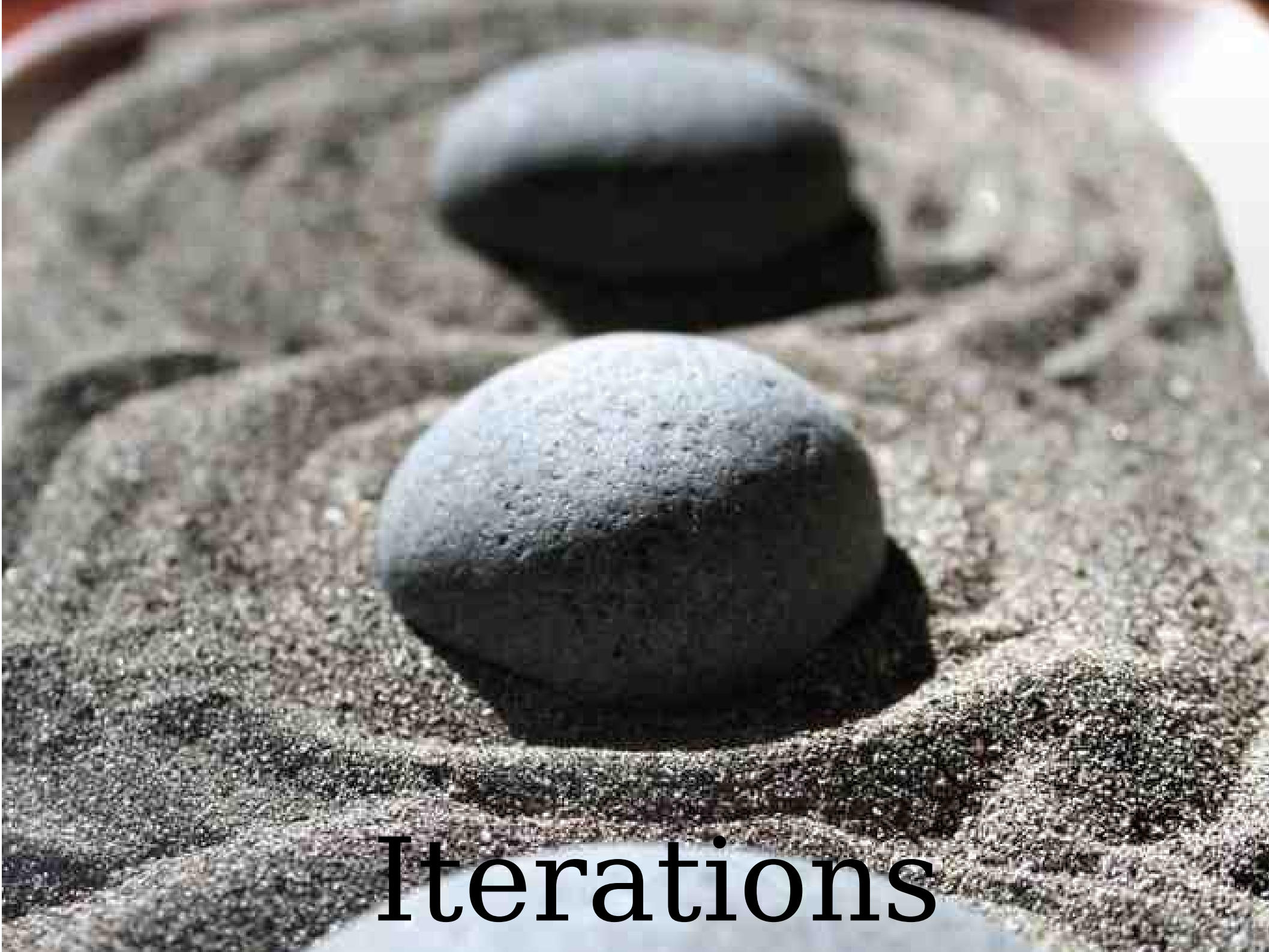
Why Wait For Low-Priority Features?

Ship Now.

Make Money.

Find Out What Customer Wants

And Add Those Features



Iterations

We've Seen

Iterations

Help

To Regularly Adjust Priorities

To Accurately Track Progress

They Also

Serve As Mini-Milestones

Or Mini-Releases

How Long Is An Iteration?

Usually

1 or 2 weeks

1 Week

Faster Feedback

More Flexible

2 Week

Coarse Grained Features

More Interdependencies With Other Disciplines

Pick What Sounds Right

You Can Always Change

1 ⇨ 2

Sum Pairs Of Past Iterations

2 ⇨ 1

Split Past Iterations Into Halves

What Happens When We Reach The End Of An Iteration...

With Incomplete Features?

All Incomplete Features

Count As 0 Points

If

Still High Priority

Work On It Next Iteration

Points Will Average Out

How Do We Know How Much Work To Schedule For Next Iteration?

“Yesterday's Weather”

We Assume Our Velocity

Will Be Our Recent Velocity

Then

Work With Customer

Highest Priority Features

Fit In As Many As We Can

Some Teams

Use Visual Reinforcement

Sheets of Paper

Size Proportional To Points

Large Sheet Of Paper

Size Proportional to Velocity

Then Fit Features Into Iteration

Intuitive

Visual (For Visual People)

So Is That It?

Let's See

We Know

Meaning of Done

We Know

How To Estimate The Complexity Of Features

We Know

When Features Are Done

We Know

How Quickly We Get Features Done

We Know

When Project Should Be Done

We Know

What Our Options Are To Change When It's Done

We Know

The Success Of Those Changes

We Know

To Make Constant Corrections

SO

Yeah.

We're

Done.

This Presentation is Licensed Under a Creative Commons 3.0 Attribution, Share-Alike License.

You may use all or part of this presentation for whatever you want, as long as you
(1) credit Michael Karlesky and Mark VanderVoord
(2) release any derived works under a similar license

http://www.creativecommons.org/

Images By (thanks!):

(cc) BY

Beach Zen - Tanais
Costa Rica Zen Garden – Scott Robinson
Fire And Water – Paul Sapiano

(cc) BY-NC

Zen Garden – Timothy Tak
Mad Scientist - Zoomar

(cc) BY-NC-SA

Momento Zen – Silke Gerstenkorn
Zen – Pierangelo Rosati
Carnival Cruise – Josh Bousel