

# Designing for Optimal Performance in SharePoint Online

*@spmcdonough  
on Twitter  
(for heckling  
purposes)*



Sean P. McDonough  
Microsoft MVP  
Chief Technology Officer  
Bitstream Foundry LLC



# Our Agenda

- SharePoint Online (SPO) Implementation
- Acknowledging the Reality of Plumbing
- SharePoint Online Diagnostics and Tools
- Design and Development Guidance
- Samples and Examples
- Questions and Answers Throughout!
- References



But first ...





An important note

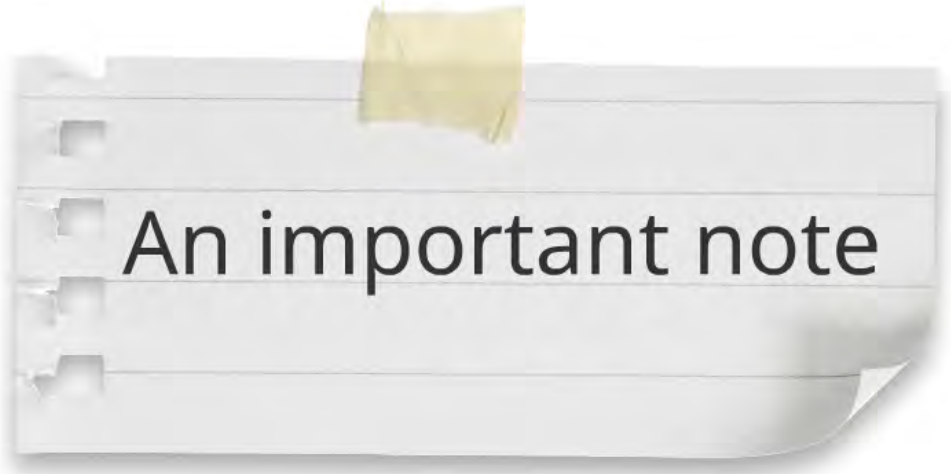


This is  
Office 365



changing and updating it"

Please don



**Office 365 is an  
"evergreen service"**



meaning "Microsoft is always changing and updating"



**Office 365 is an  
"evergreen service"**



meaning "Microsoft is always changing and updating it"

**What I show you today ...**

- will probably be true tomorrow



# always changing and updating

## What I show you today ...

- will probably be true tomorrow
- has a good chance of being true next week
- might be true in month
- probably worth questioning and re-evaluating in a year





Please don't dig this up in five years and then send me hate mail because I presented something that is no longer accurate due to a SharePoint Online service change.



Please don't dig this up in five years and then send me hate mail because I presented something that is no longer accurate due to a SharePoint Online service change.



**Dear Sean,**

I was reviewing a presentation you put together five years ago, and I found elements that were incorrect. You are a horrible person and you should never touch SharePoint Online again.

Love you lots!  
- an attendee





# First Stop:

Some basic  
SharePoint farm  
architecture

(and why  
that matters  
with SPO)



Welcome to the farm!

# You might think that SPO is simply an extension of this pattern

On-premises SharePoint farms come in all shapes and sizes

## Small Farm

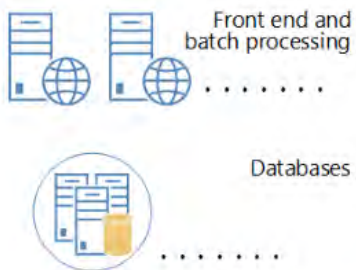
<1,000 users

Fault tolerance for simple workloads with small volumes of content

Two tiers:

- Combined front-end and batch processing servers
- Database servers

Scale the number of servers as needed.



## Medium Farm

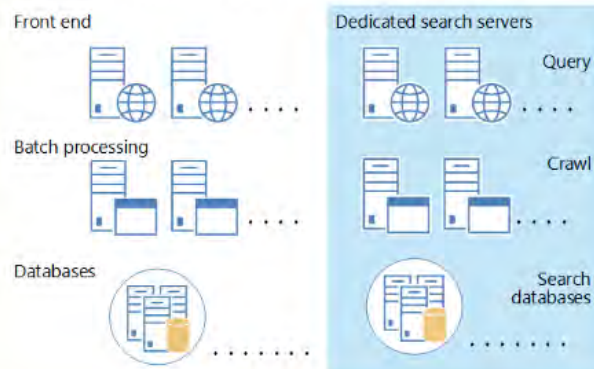
<10,000 users

Dedicated search servers for up to 10 million items.

Three tiers:

- Front-end servers
- Batch processing servers
- Database servers

Scale the number of servers as needed.

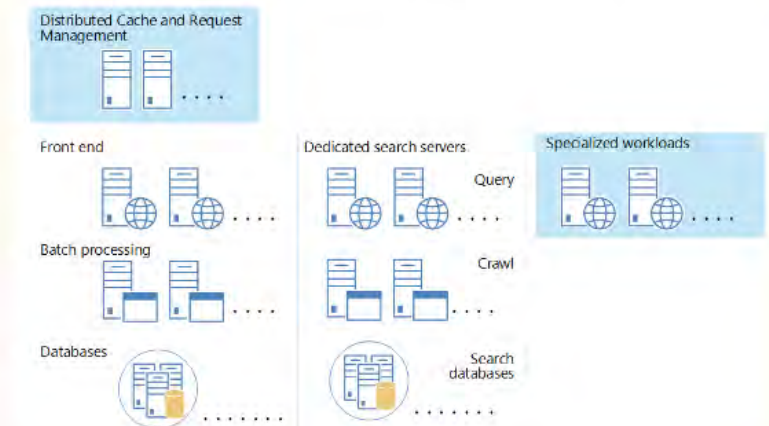


## Large Farm

More than 10,000 users

Additional server types to support large farms.

This farm represents each of the server roles that are recommended. For each server role the servers are configured identically. Scale each server role independently. Large farms benefit by adding dedicated servers for Distributed Cache and by adding Request Management.



Databases



Search databases

well, not really ...

You might think that SPO is simply an extension of this pattern.

On-premises SharePoint farms come in all shapes and sizes

## Small Farm

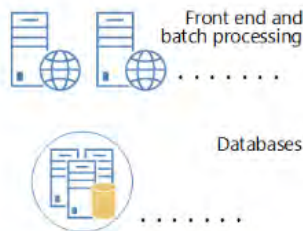
<1,000 users

Fault tolerance for simple workloads with small volumes of content

Two tiers:

- Combined front-end and batch processing servers
- Database servers

Scale the number of servers as needed.



## Medium Farm

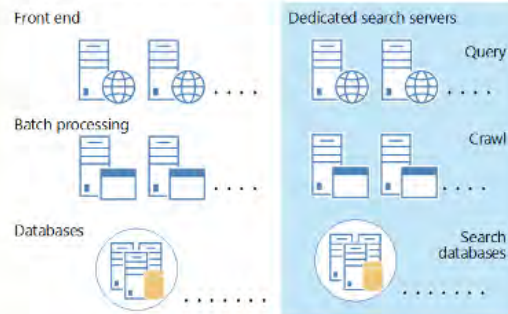
<10,000 users

Dedicated search servers for up to 10 million items.

Three tiers:

- Front-end servers
- Batch processing servers
- Database servers

Scale the number of servers as needed.

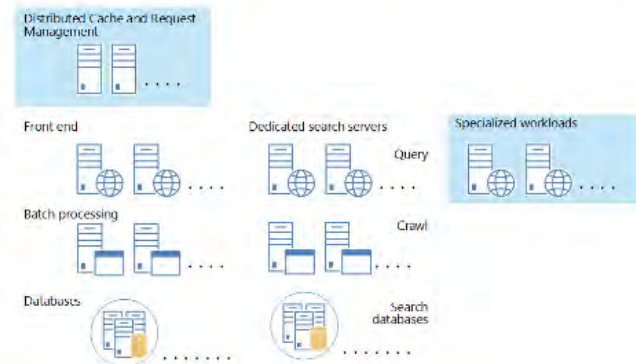


## Large Farm

More than 10,000 users

Additional server types to support large farms.

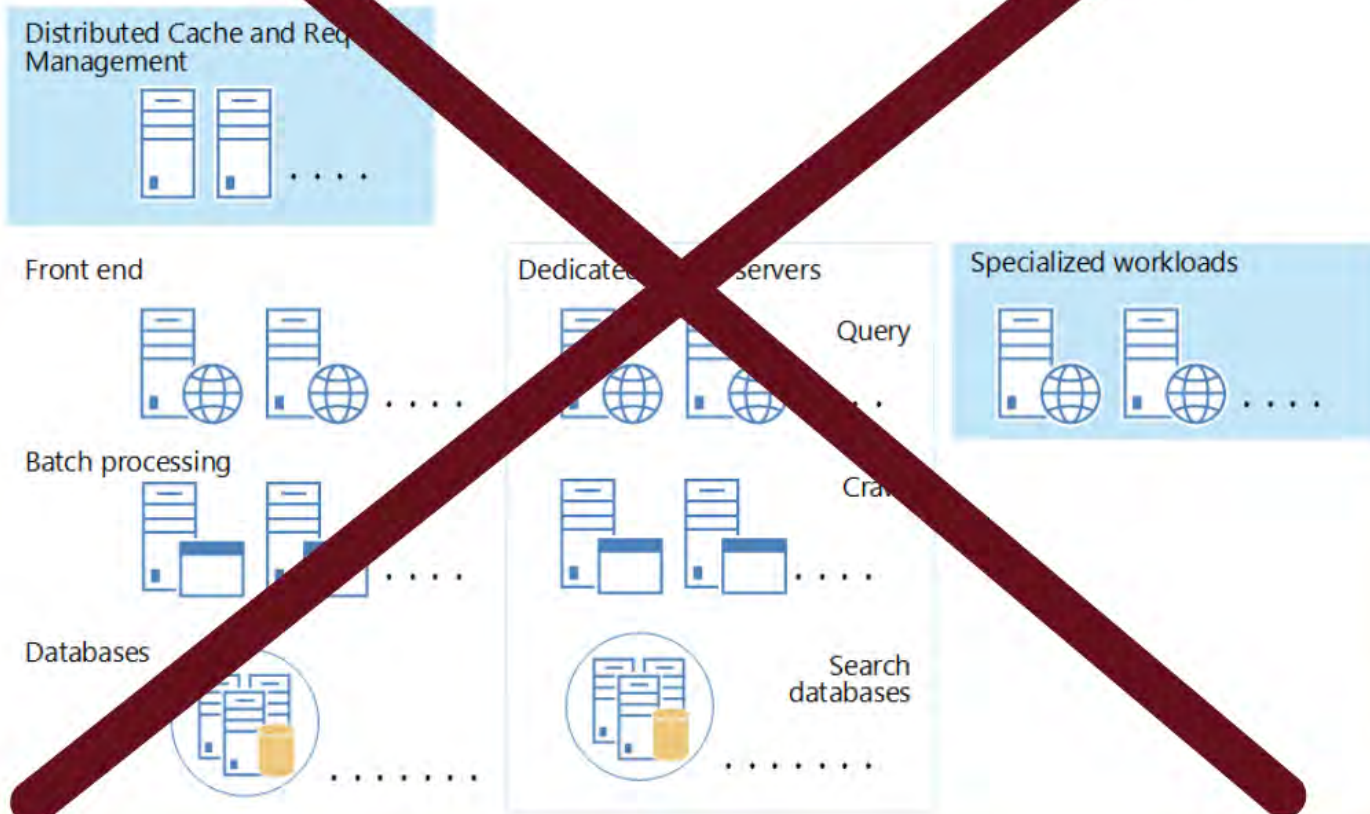
This farm represents each of the server roles that are recommended. For each server role the servers are configured identically. Scale each server role independently. Large farms benefit by adding dedicated servers for Distributed Cache and by adding Request Management.



More than 10,000 users

Additional server types to support large farms.

This farm represents each of the server roles that are recommended. For each server role the servers are configured identically. Scale each server role independently. Large farms benefit by adding dedicated servers for Distributed Cache and Request Management.



It is not.

well, not really ...

You might think that SPO is simply an extension of this pattern.



This is a stamp  
too (well covered of them)





This is a stamp  
too (well, several of them)

Datacenter 1..N:

Network 1..N:

Disaster Recovery Datacenter 1..N:

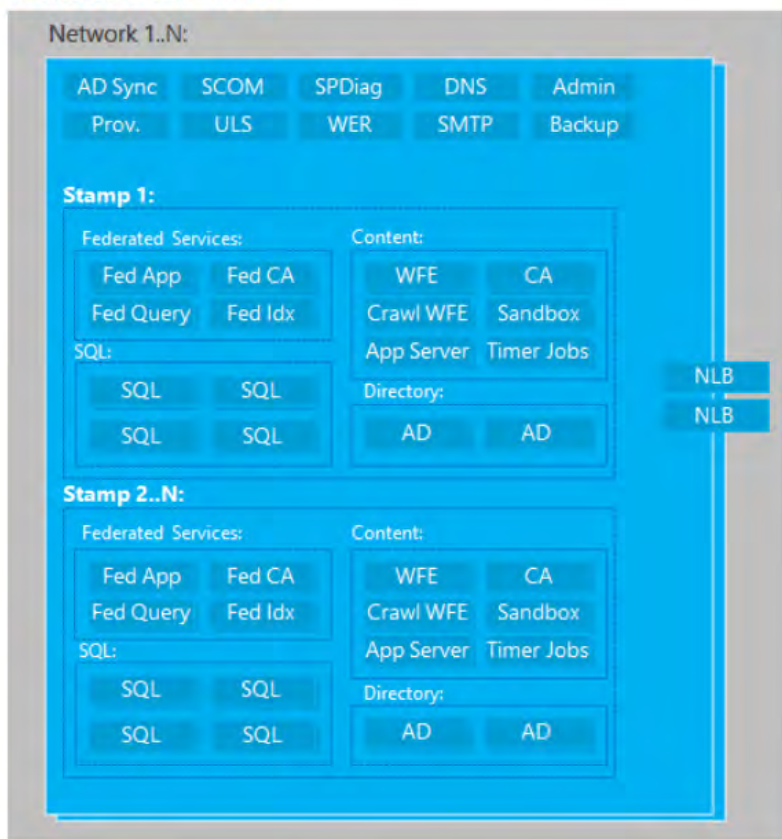
Network 1..N:

Grid Manager

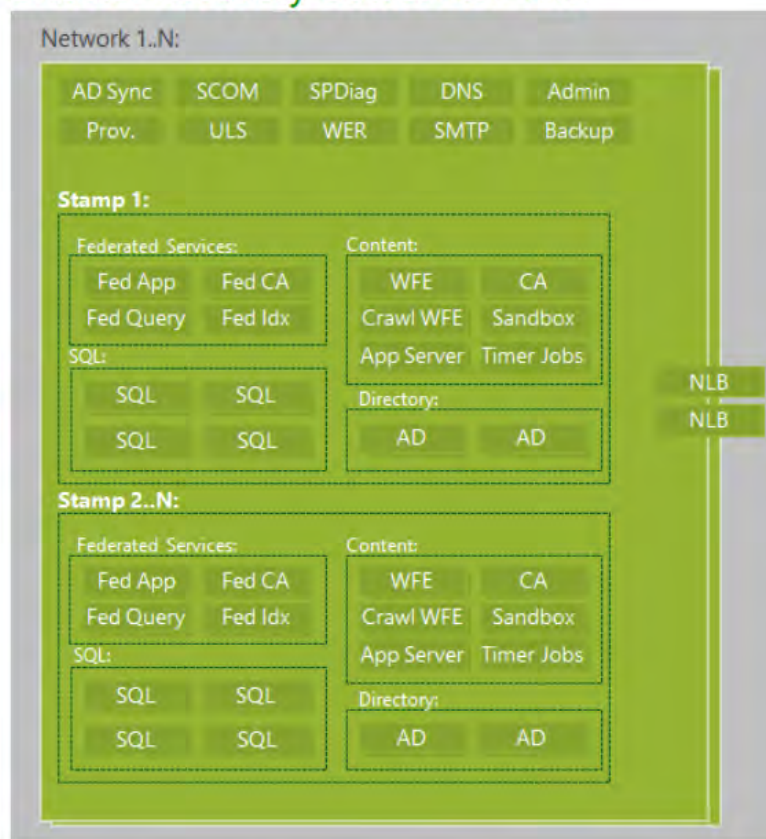


# This is a stamp too (well, several of them)

Datacenter 1..N:



Disaster Recovery Datacenter 1..N:



- Grid Manager
- Global Directory
- Tenant Admin (UI)
- Commerce backend
- DNS (multiple)
- OrgID Auth, Svc.
- Incident Management
- Azure (Windows/SQL)
- CDN Services

Looking at the representation of an individual stamp, you might think it's only 16 servers.

Are you  
ready for



THIS  
too (well,

Each datacenter has two or more stamps per SPO environment for high-availability.

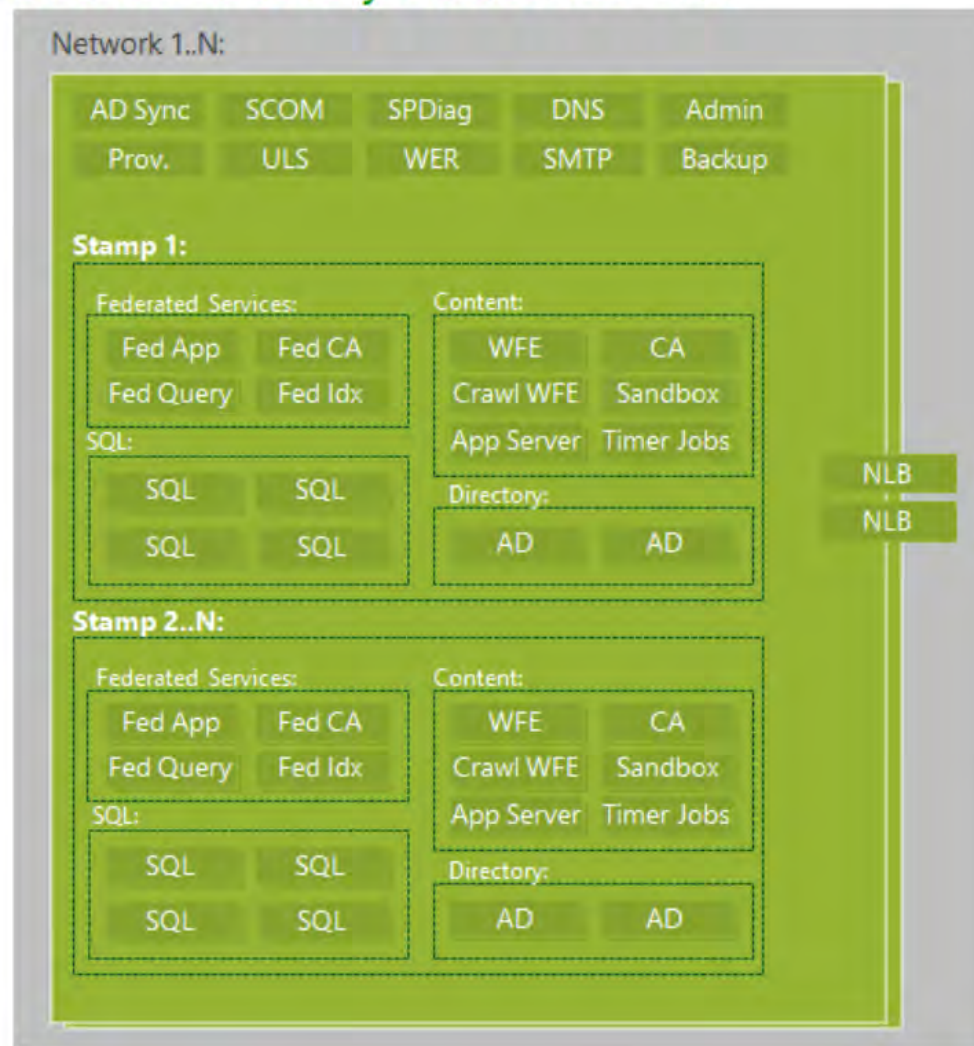
### Datacenter 1..N:



Looking at the representation of an individual stamp, you might think it's only 16 servers.

# s is a stamp (several of them)

## Disaster Recovery Datacenter 1..N:



- Grid Manager
- Global Directory
- Tenant Admin (UI)
- Commerce backend
- DNS (multiple)
- OrgID Auth, Svc.
- Incident Management
- Azure (Windows/SQL)
- CDN Services

Additional stamps exist in a different region for redundancy and failover.

Are you





Looking at the representation of an individual stamp, you might think it's only 16 servers.

Are you  
ready for  
the kicker?





The exact number of servers in a SharePoint Online stamp is variable.

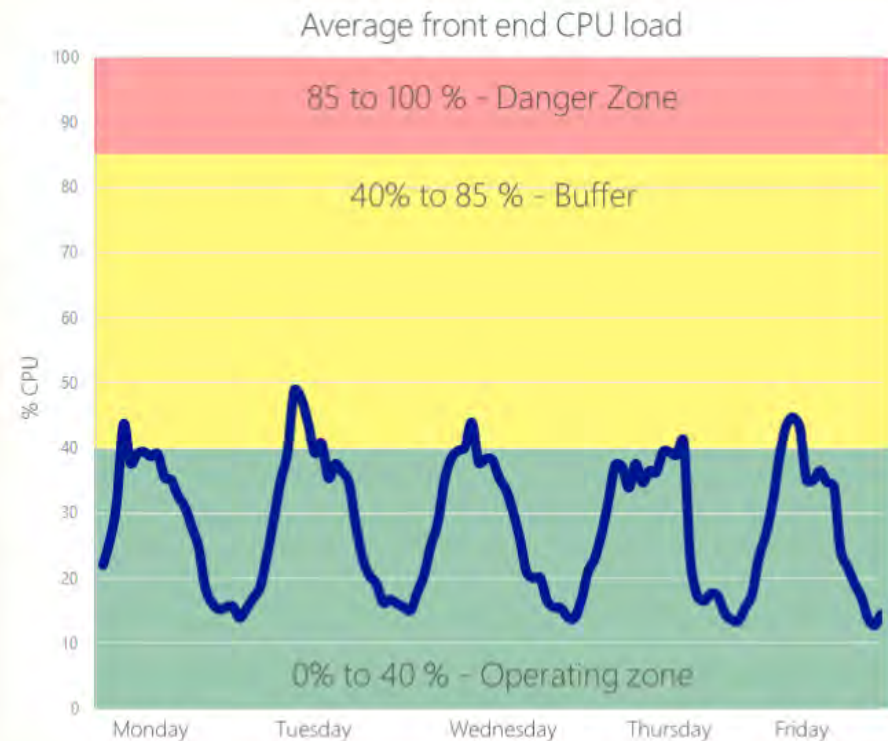
The number of servers per





# The exact number of servers in a SharePoint Online stamp is variable.

The number of servers per stamp varies because the server count is adjusted based on average front-end CPU load.

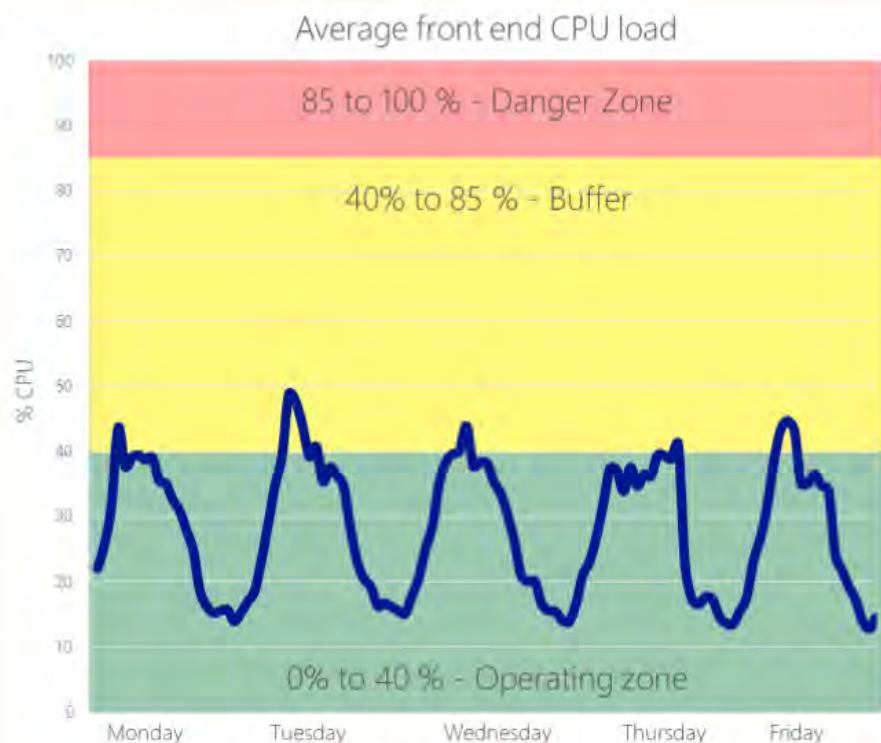


- If load rises above 40%\*, additional servers are automatically provisioned and added to the stamp



# SharePoint Online stamp is variable.

The number of servers per stamp varies because the server count is adjusted based on average front-end CPU load.



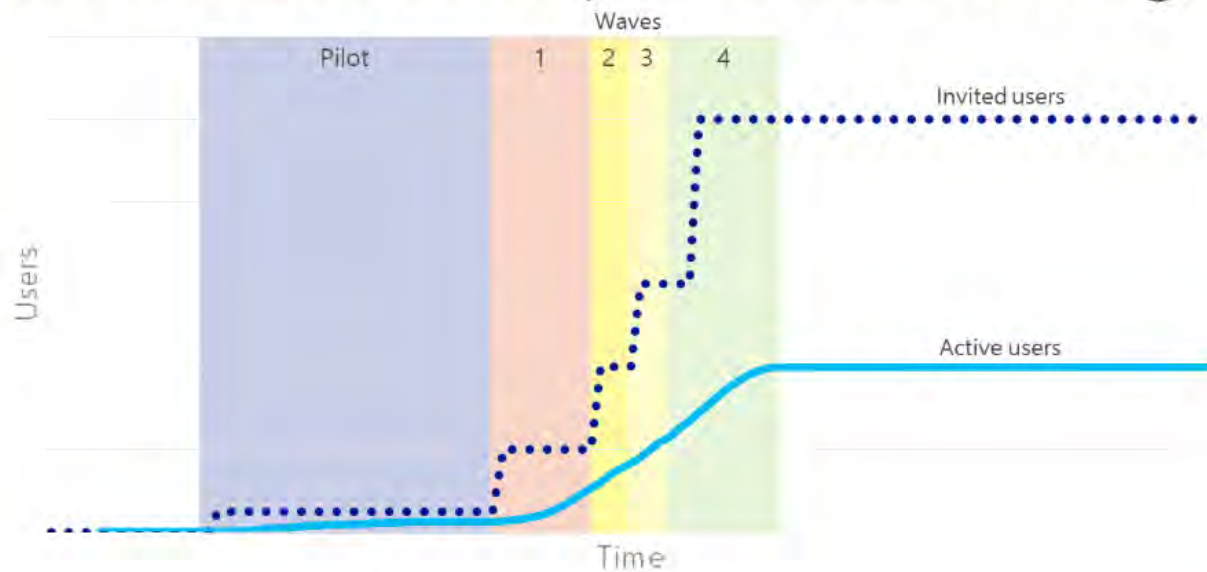
- If load rises above 40%\*, additional servers are automatically provisioned and added to the stamp.
- If load drops, servers are decommissioned.

# Adding and removing is not an instantaneous process, though.



an control load (for example, gradually increasing the number of users over time when you roll out a site) you should It w

# Adding and removing is not an instantaneous process, though.



If you can control load (for example, gradually increasing the number of invited users over time when you roll out a site), you should. It will give the provisioning system time to adjust/compensate for growing load.

If you remember only one thing in this discussion of stamps and elastic capacity, please let it be this one point ...





Load  
testing  
is futile.

astic nature of a stamp, there's really no way to effectively load t

g the number of  
uld. It will give  
growing load.



please let it be this one point...

# Load testing is futile.

- Given the elastic nature of a stamp, there's really no way to effectively load test SPO. Any numbers you get or produce are essentially meaningless in the grand scheme of things.
- Rather than load testing, focus instead on the items we're going to cover in the rest of this presentation. They'll help you avoid poorly performing pages and sites.

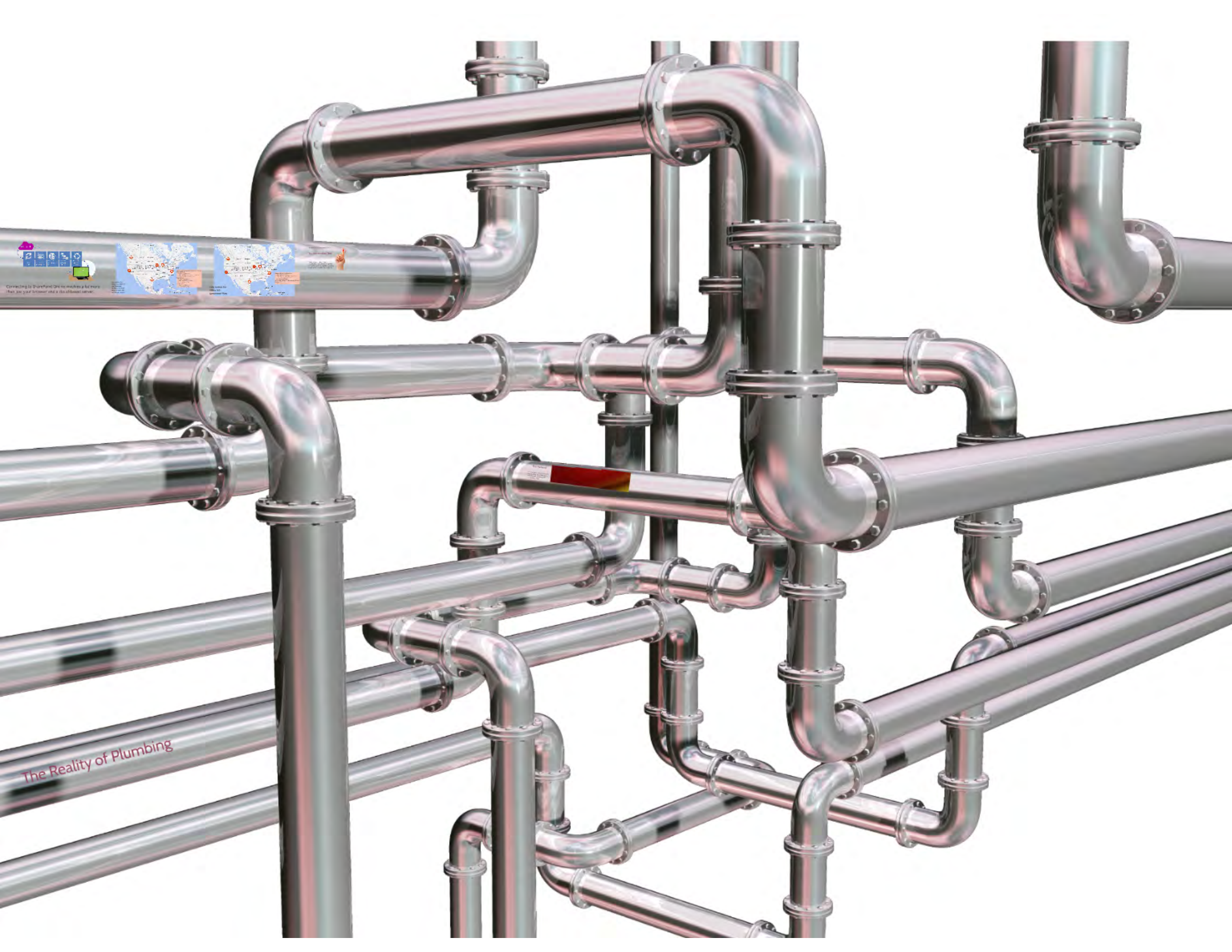


Welcome to the farm!



# The Reality of Plumbing





Connected to SharePoint Online to make it easier for you to find the information you need. For more information, visit [sharepoint.com](#).

The Reality of Plumbing

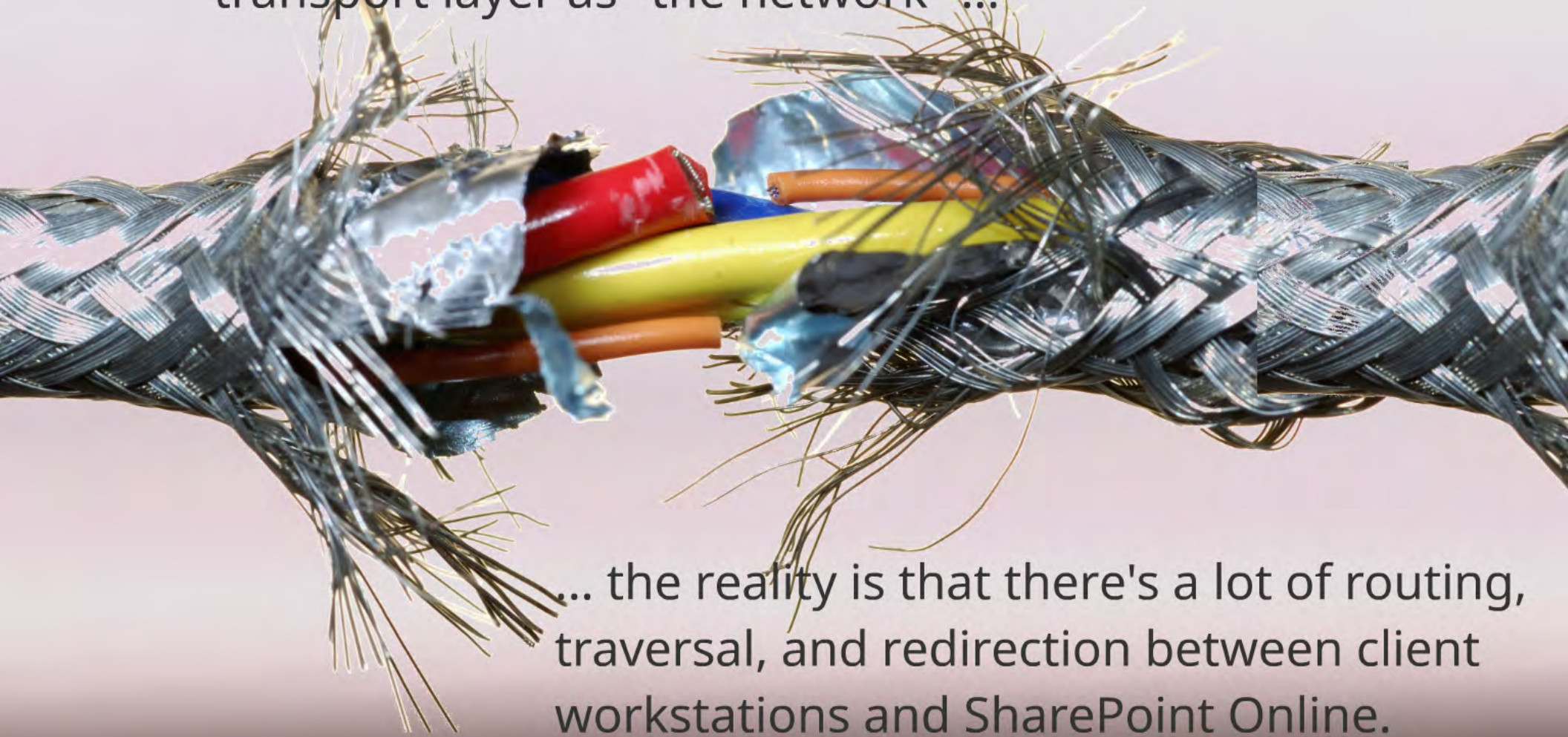
# The Network

# The Network

If we're talking about end-to-end performance, we really need to start with the plumbing that carries everything.



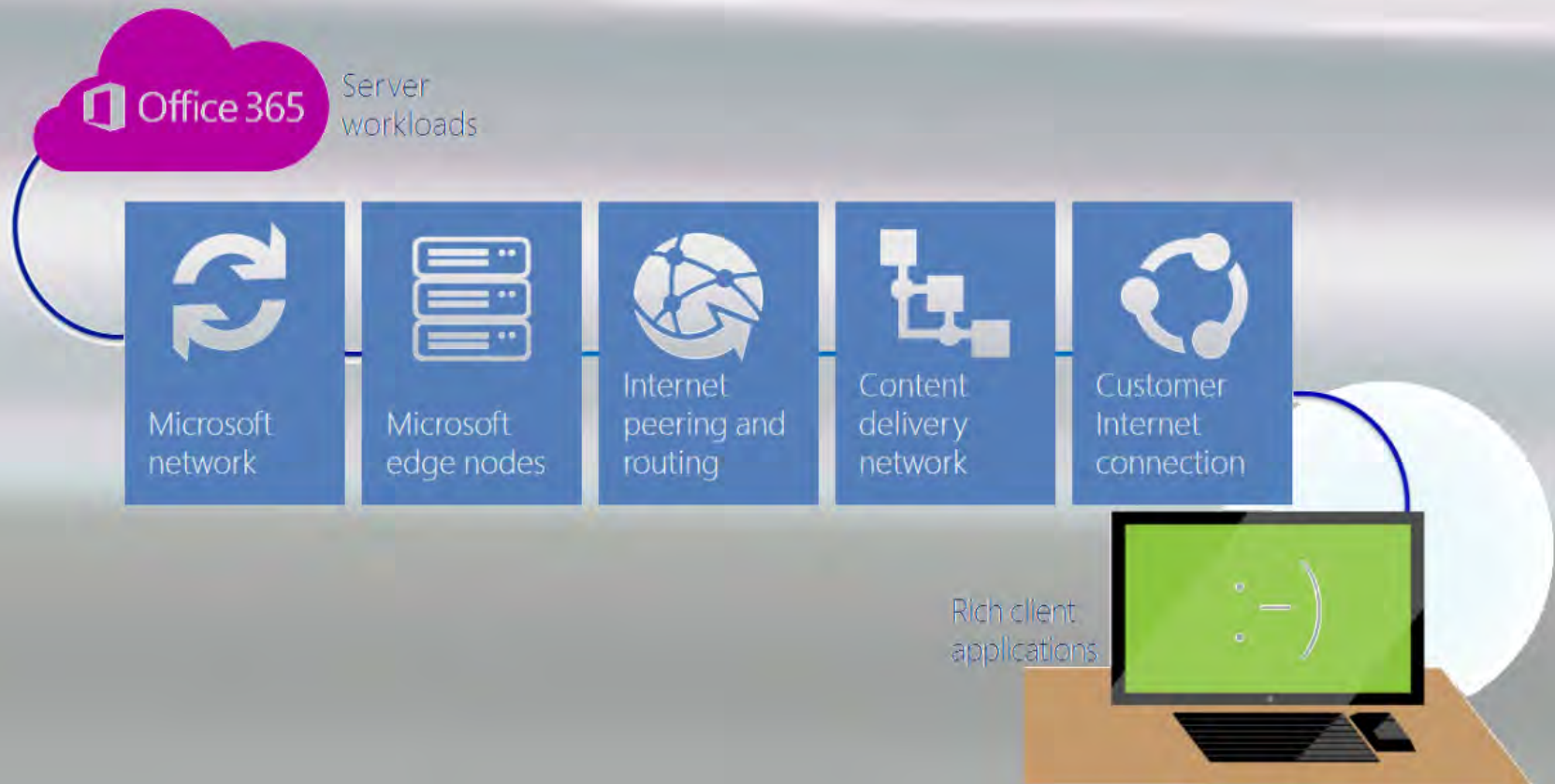
Although it's easy to abstractly refer to the transport layer as "the network" ...



... the reality is that there's a lot of routing, traversal, and redirection between client workstations and SharePoint Online.



If the network is "out of whack,"  
it can look like SharePoint  
Online itself is the issue.



Connecting to SharePoint Online involves a lot more than just your browser and a cloud-based server.

We don't have time to cover all the plumbing in this session ...

So, remember this

If you've spent a lot of time

We don't have time to cover all the plumbing in this session ...

## So, remember this:

If you've spent a lot of time troubleshooting in SharePoint Online (to little or no effect), maybe you should zoom out and consider the network.





Okay, so it really feels like there is a tangible performance problem.

Okay, so it really feels like there is a tangible performance problem.



How do you prove it objectively?



The good news: the destruction of computer equipment is not necessary (although you might feel compelled to do so)!

objectively?

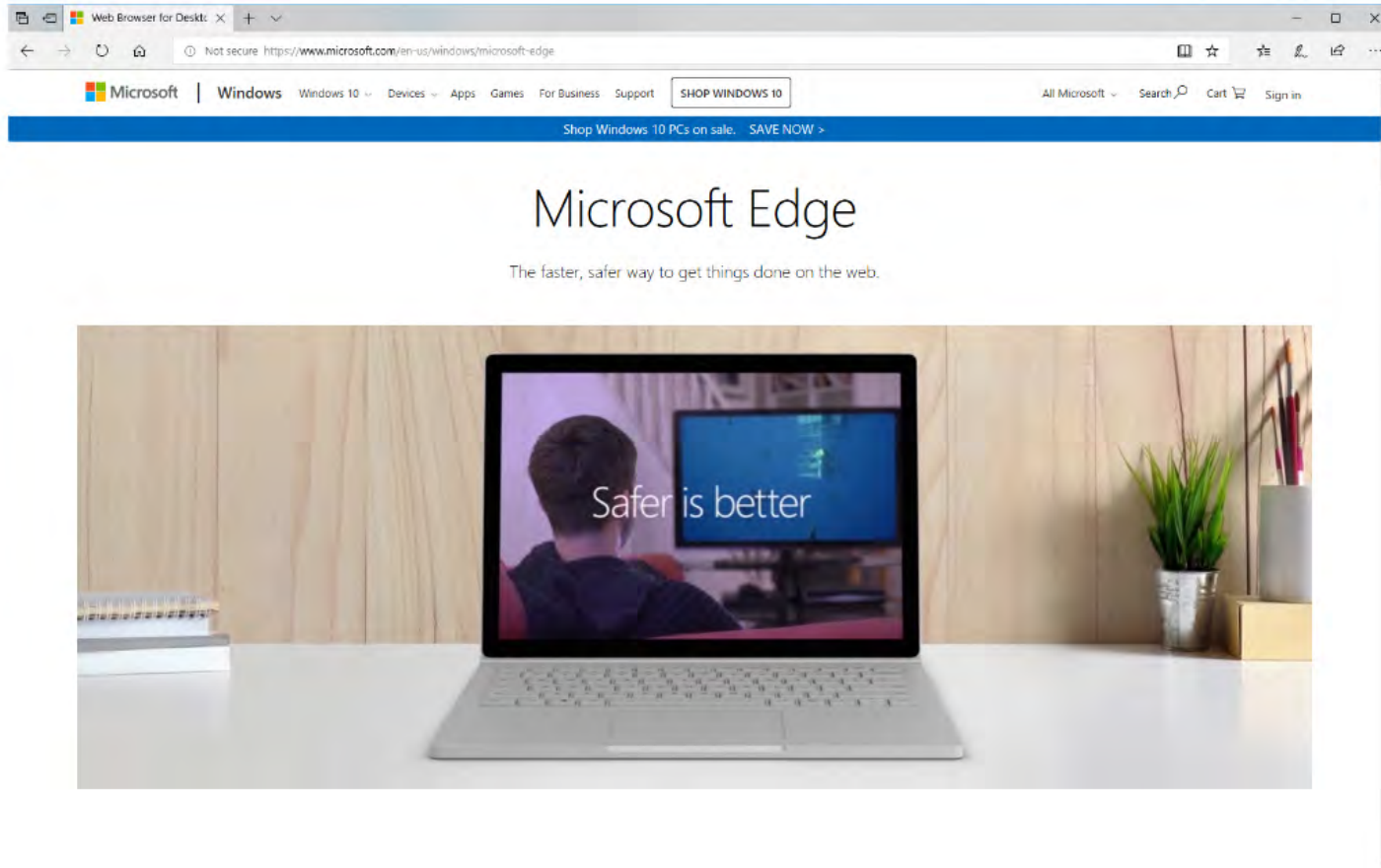
Meet your primary  
diagnostic tool.



Meet your primary  
diagnostic tool.



In all likelihood, you already  
have it on your system.



# Hello, Edge!



like



Um ...  
you're  
kidding,  
right?

None!



Um ...  
you're  
kidding,  
right?

Nope!



Bitstream Foundry LLC

Bitstream Foundry LLC

☆ Not following Share

🔍 Search

📁 Clients

📄 Pages

🗑️ Recycle bin

✎ Edit

+ New ✎ Quick edit 📄 Export to Excel 📄 Flow 📄 Power Apps 📄 All Items 📄 🔍 ⓘ

## Client Projects

Attachments Title Description Client

DEMO

[Return to classic SharePoint](#)

Feedback

- \* may be due to routing issues (as in "number of hops")
- \* plenty of other possibilities

# Demo Takeaways

HTTP

Response

Headers

waiting on server -  
generally zero or  
near zero

time spent  
processing on  
server (in ms)  
- ideally low

- SPIisLatency
- SPRequestDuration
- X-SharePointHealthScore — 0 to 10  
(you want 0)

Don't see the headers?

Don't panic!

I'll be demonstrating a better way to get the performance numbers (and site assessments) a little bit later!

# HTTP Response Headers

- SPIisLa

## Generally speaking ...

Name / Path	Protocol	Method	Result / Description	Content type	Received	Time	Initiator / Type	0m	Headers	Body	Parameters
Home.aspx https://cardinal...	HTTPS	GET	200 OK	text/html	75.36 KB	2.01 s	document		SPIisLatency: 1		
									SPRequestDuration: 1456		

$\text{Time} - (\text{SPRequestDuration} + \text{SPIisLatency}) = \text{"time lost elsewhere"}$

- \* potential network latency
- \* may be due to routing issues (as in "number of hops")
- \* plenty of other possibilities

# Demo Takeaways

HTTP

Response

waiting on server -  
generally zero or

time spent  
processing on

So, you've concluded that your pages are slow and you have the data to prove it!

SPIisLatency is low, and  
X-SharePointHealthScore is low,

but ...

So, you've concluded that your pages are slow and you have the data to prove it!

SPIisLatency is low, and  
X-SharePointHealthScore is low,

but ...

**SPRequestDuration is crazy high (e.g., 9000 ms)!**



Repeat after me ...

"The problem probably isn't

# Repeat after me ...

"The problem probably isn't  
SharePoint Online. It's my site."





Okay, one more time:

Repe

"The pro  
SharePoir

# Repeat after me ...

"The problem probably isn't  
SharePoint Online. It's my site."

So, who's to blame?

In all likelihood

So, who's to blame?

In all likelihood:  
blame the  
lousy\* devs.



\*Note: not all devs are lousy devs. Just the ones who create performance problems and knee-jerk into blaming Microsoft and SharePoint Online.

# So, who's to blame?

In all likelihood:  
blame the  
lousy\* devs.



\*Note: not all devs are lousy devs. Just the ones who create performance problems and knee-jerk into blaming Microsoft and SharePoint Online.

- Compare processing and response times for a SharePoint site or page. ← (may



- In the majority of poor performance scenarios, a combination of UI/UX , client-side code additions, and questionable customization/deployment mechanisms are to blame.
- Microsoft has indicated that the slowest 1% of pages in SPO take more than 5,000ms to load - again, usually due to customizations.

...nes who create performance  
...t and SharePoint Online.

- Compare processing and response times to your problematic SharePoint site or page.  (may not be valid approach much longer ...)

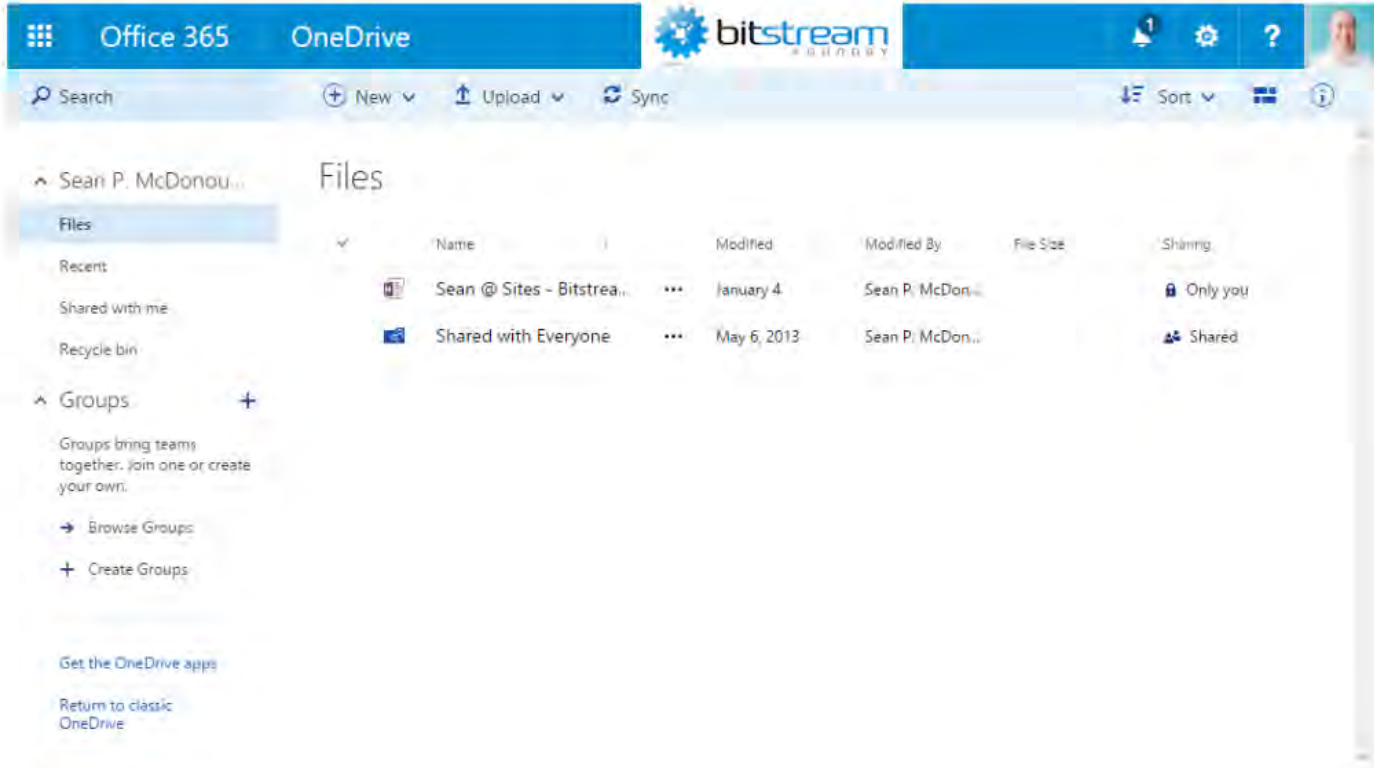
- In the majority of poor performance scenarios, a combination of UI/UX , client-side code additions, and questionable customization/deployment mechanisms are to blame.

- Microsoft has indicated that the slowest 1% of pages in SPO take more than 5,000ms to load - again, usually due to customizations.

Don't believe me?



# Collect the data and validate for yourself!



- Profile your OneDrive for Business page (it's in your MySite).
- Compare processing and response times to your problematic SharePoint site or page. *(may not be valid approach much longer ...)*

• In the majority of poor performance scenarios a

## Don't believe me?



"Okay, yeah - my OneDrive for Business page is really fast ... but my SharePoint pages are completely choking."

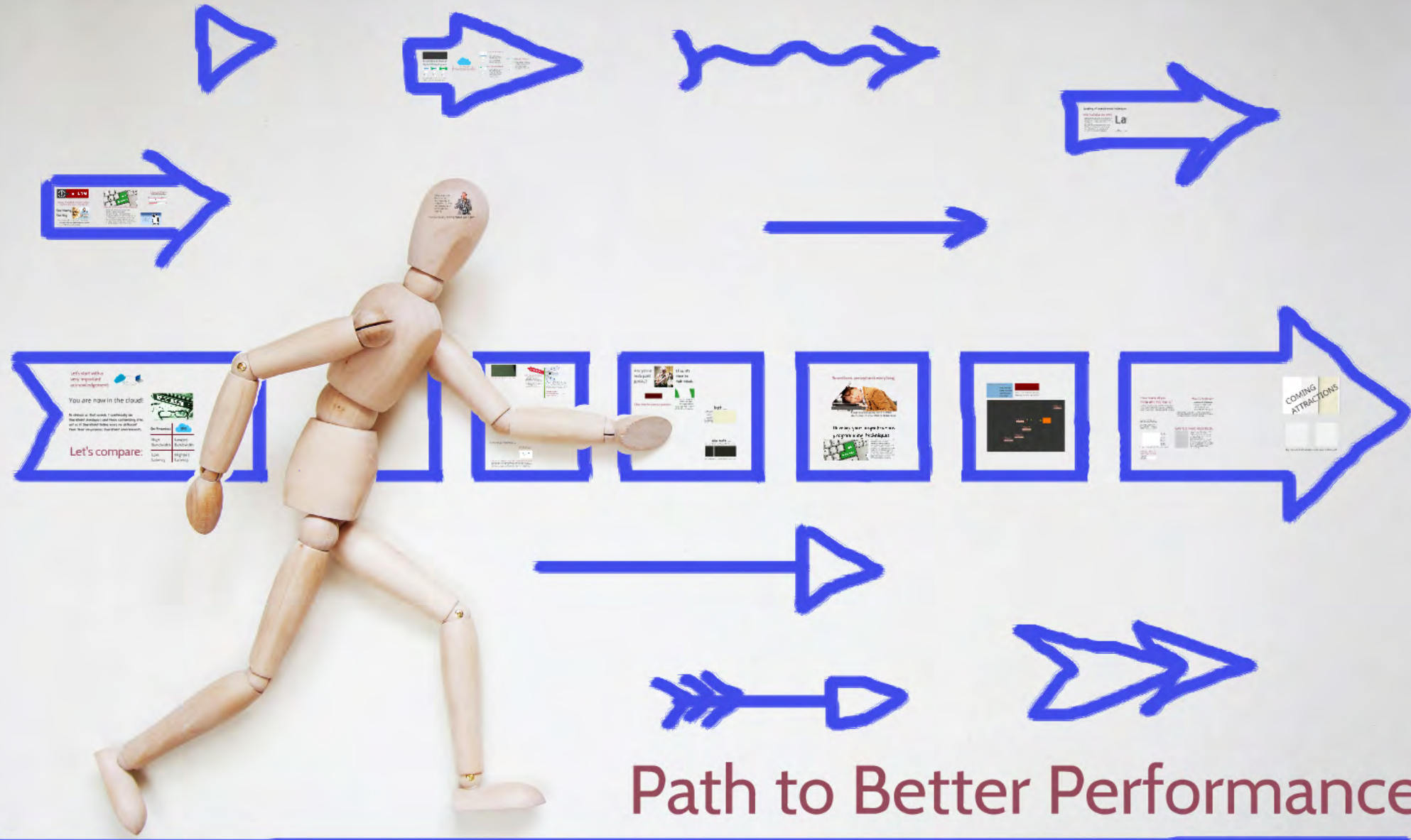


're probably thinking "What can I do

"Okay, yeah - my OneDrive for Business page is really fast ... but my SharePoint pages are completely choking."



You're probably thinking **"What can I do?"**



# Path to Better Performance

Let's start with a very important architectural requirement

You are now in the cloud!

It allows us that much, I'm talking on the cloud. It allows us that much, I'm talking on the cloud. It allows us that much, I'm talking on the cloud.

Let's compare:

High	Low
Low	High
High	Low
Low	High

Let's start with a  
very important  
acknowledgement:

Let's start with a  
very important  
acknowledgement:



You are now in the cloud!

As obvious as that sounds, I continually see

acknowledgement:

# You are now in the cloud!

As obvious as that sounds, I continually see SharePoint developers and those customizing sites act as if SharePoint Online were no different than their on-premises SharePoint environments.

very important  
acknowledgement:




# You are now in the cloud!

As obvious as that sounds, I continually see SharePoint developers and those customizing sites act as if SharePoint Online were no different than their on-premises SharePoint environments.

## Let's compare:



On-Premises	 SPO
High Bandwidth	Low(er) Bandwidth
Low Latency	High(er) Latency



Failing to acknowledge the "we're in the cloud now" reality leads to a problem I simply call ...



Too Many



Failing to acknowledge the "we're in the cloud now" reality leads to a problem I simply call ...

Too Many,  
Too Big



- Too many calls are made to the server.

now" reality leads to a problem I simply call ...

# Too Many, Too Big



- Too many calls are made to the server.
- Too many files are referenced on pages.
- The files in-use are too large.



Consider one or more of the following:

- Minify files, especially JavaScript files.

• Resize images to usage sizes

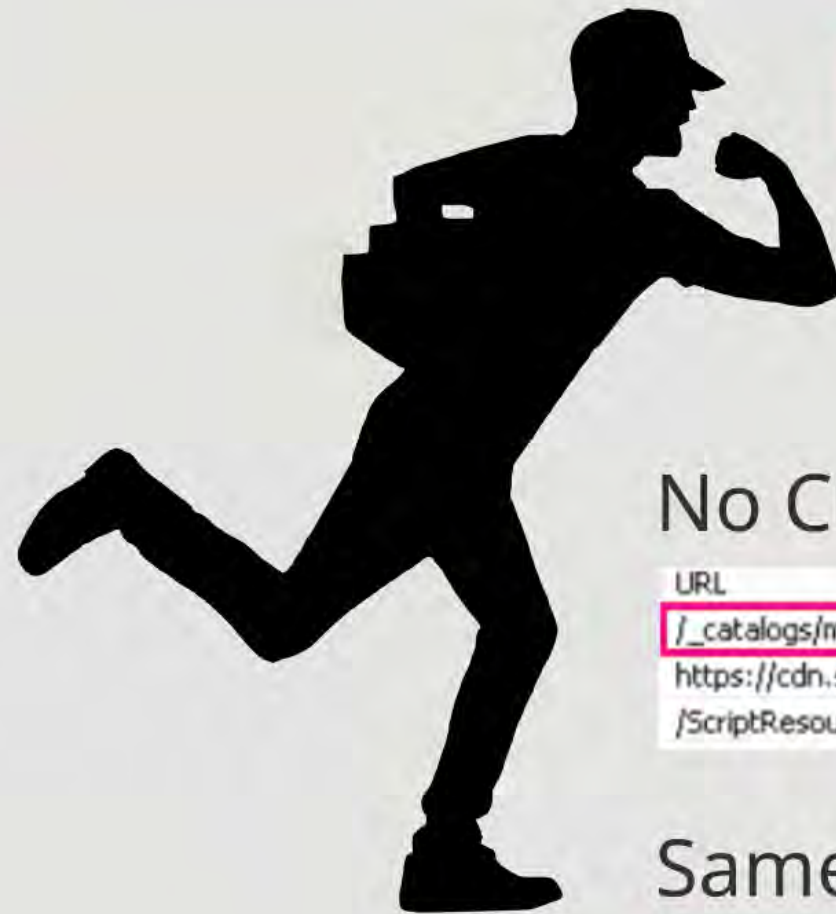


## Consider one or more of the following:

- Minify files, especially JavaScript files.
- Resize images to usage sizes.
- Compress images (more) aggressively.
- Use sprite sheets to reduce the actual number of HTTP requests needed to retrieve images.
- Use SharePoint's Image Rendition service.
- Leverage a toolkit like Font Awesome in place of individual icons and associated files.

And the big  
kahuna ...





# Use a CDN!!!

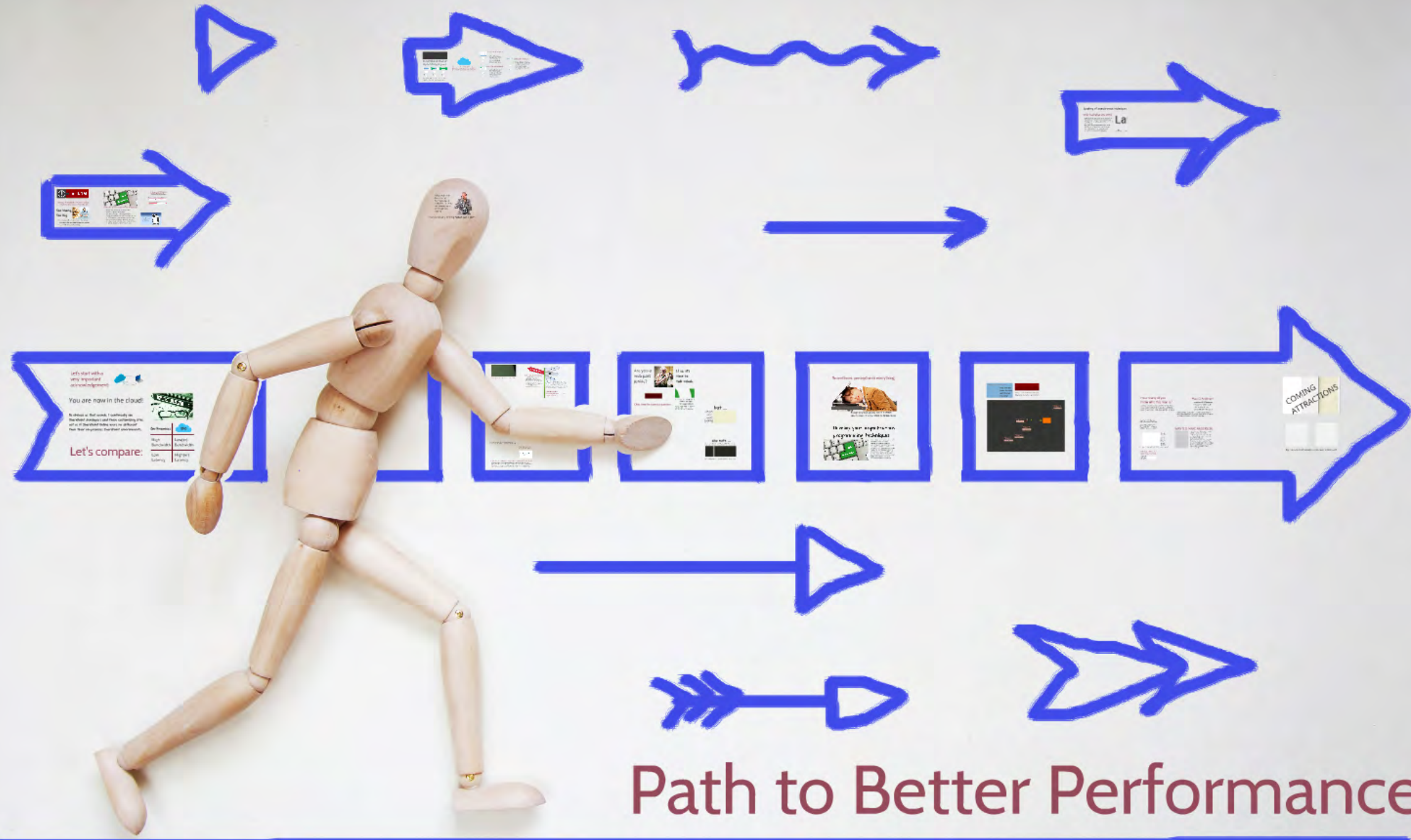
(Content Delivery Network)

No CDN in use (i.e., SPO library direct)

URL	Received	Taken	Initiator
<a href="#">/_catalogs/masterpage/javascript/jquery-2.1.1.min.js</a>	82.98 KB	1.51 s	<script>
<a href="https://cdn.sharepointonline.com/12413/_layouts/15/16">https://cdn.sharepointonline.com/12413/_layouts/15/16</a>	18.98 KB	156 ms	<script>
<a href="#">/ScriptResource.axd?d=M1vNi_a6A2vtkOenP45i9-peGfx</a>	100.80 KB	2.04 s	<script>

Same resource from a CDN

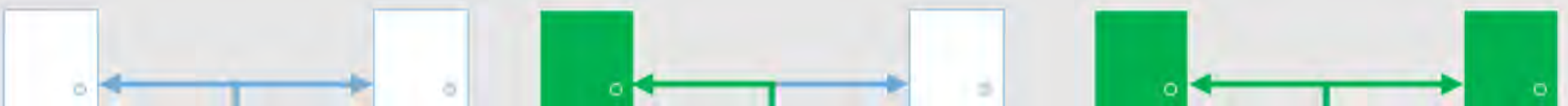
URL	Received	Taken
<a href="https://ajax.aspnetcdn.com/ajax/jQuery/jquery-2.1.1.min.js">https://ajax.aspnetcdn.com/ajax/jQuery/jquery-2.1.1.min.js</a>	82.74 KB	469 ms
<a href="#">/WebResource.axd?d=r3Mv/y4JFCBwmUs1-gLXCgVJy4RMAH/qCj2oIh3D5kbMXzSdwm5KlpDx9vM6MKjztZon...</a>	22.33 KB	0.84 s
<a href="#">/_layouts/15/images/spcommon.png?rev=38</a>	20.56 KB	1.15 s



Path to Better Performance

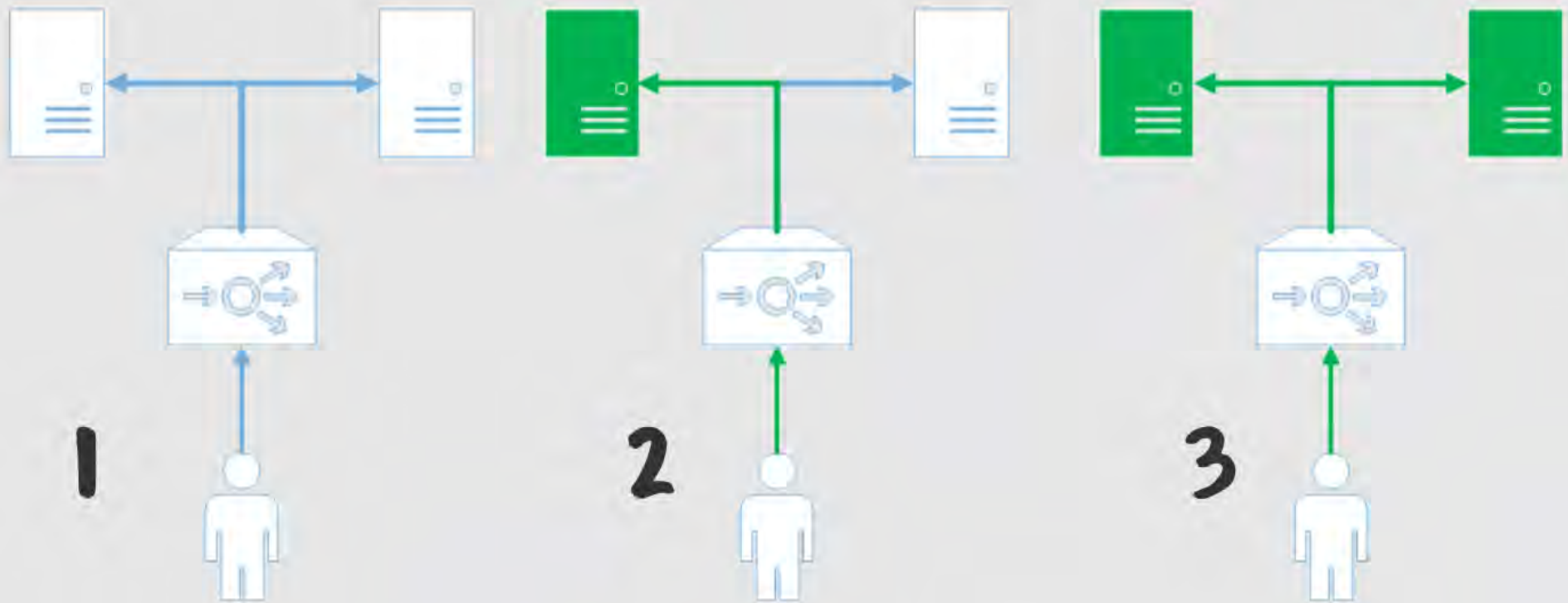


Conventional wisdom  
says caching is good.

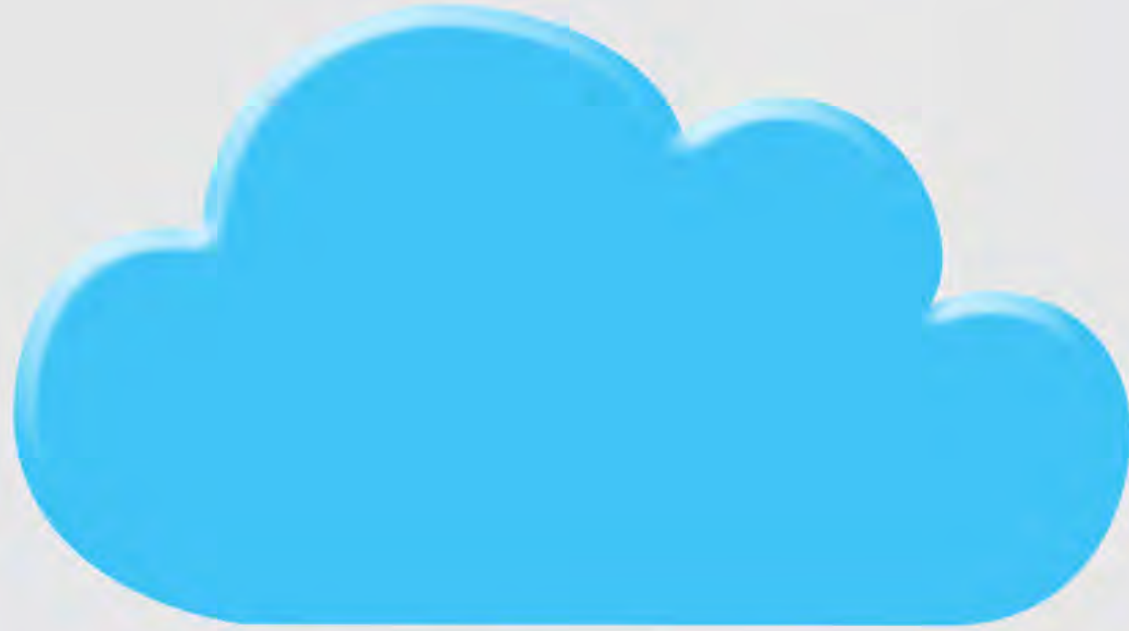




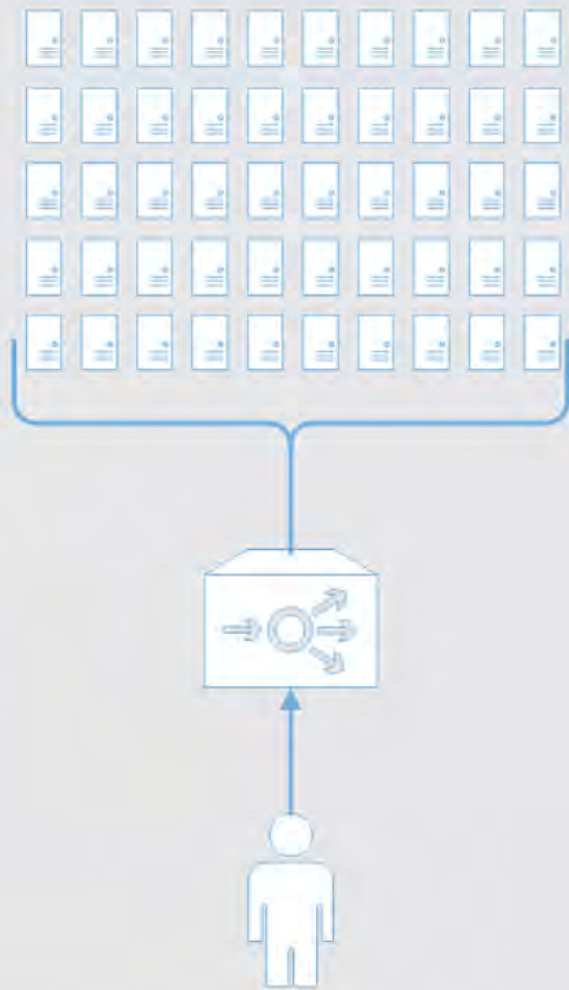
# says caching is good.



After just a few requests, the on-premises Object Cache can be "ready for action."



In the cloud, the caching equation (for per-server memory-based caches like the Object Cache) works out a bit differently.



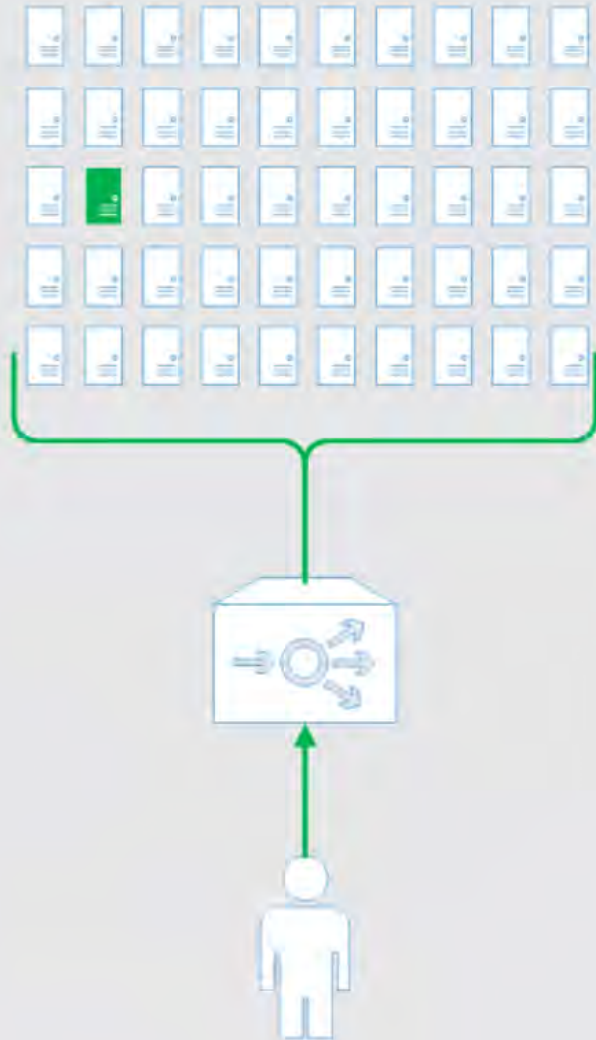
# User's Initial Request

- First thing to note: the number of WFEs tends to be \*much\* higher in the cloud versus on-premises.

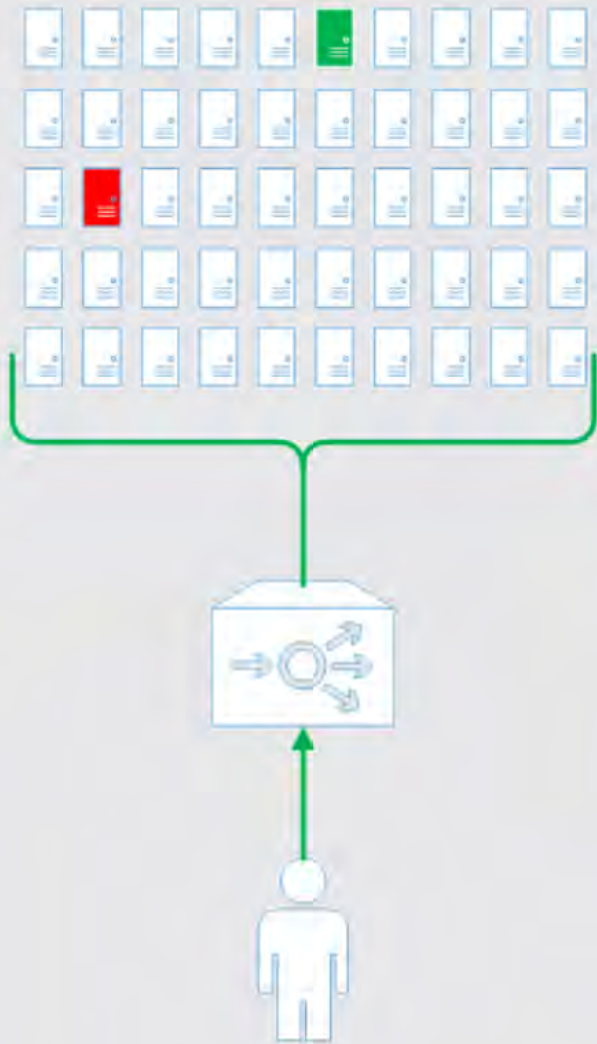


User's Second Request

# User's Second Request

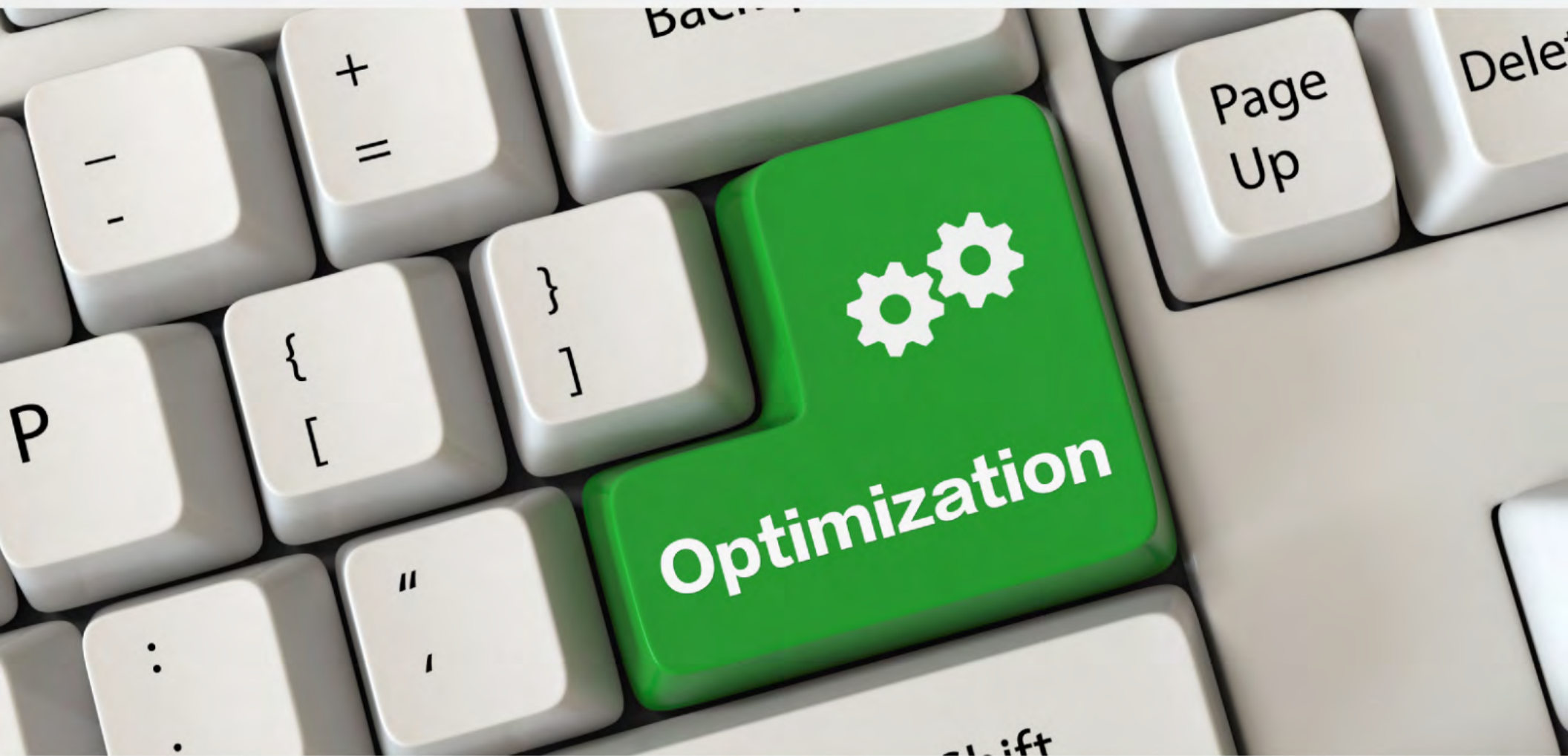


- No affinity is in use, so the chance of a user hitting the same server again is dramatically less than the on-premises scenario.



## Subsequent Requests

- Same reduced chance of hitting the WFE last visited
- Memory pressure causes much more frequent cache ejections versus on-premises.



Two significant adjustments can be made.

\* These sitemaps are then stored in the Object

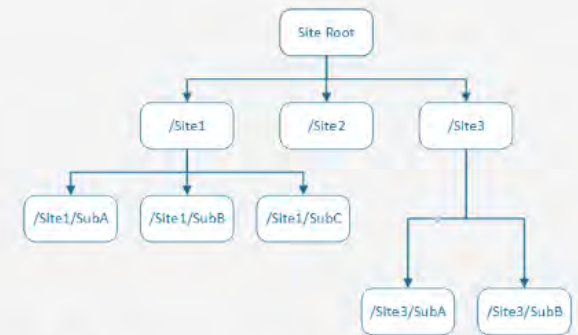
Navigation style  
has a huge impact  
on performance.

CHOOSE

Using structural navigation is the default (but poor) choice for complex site hierarchies in the cloud.

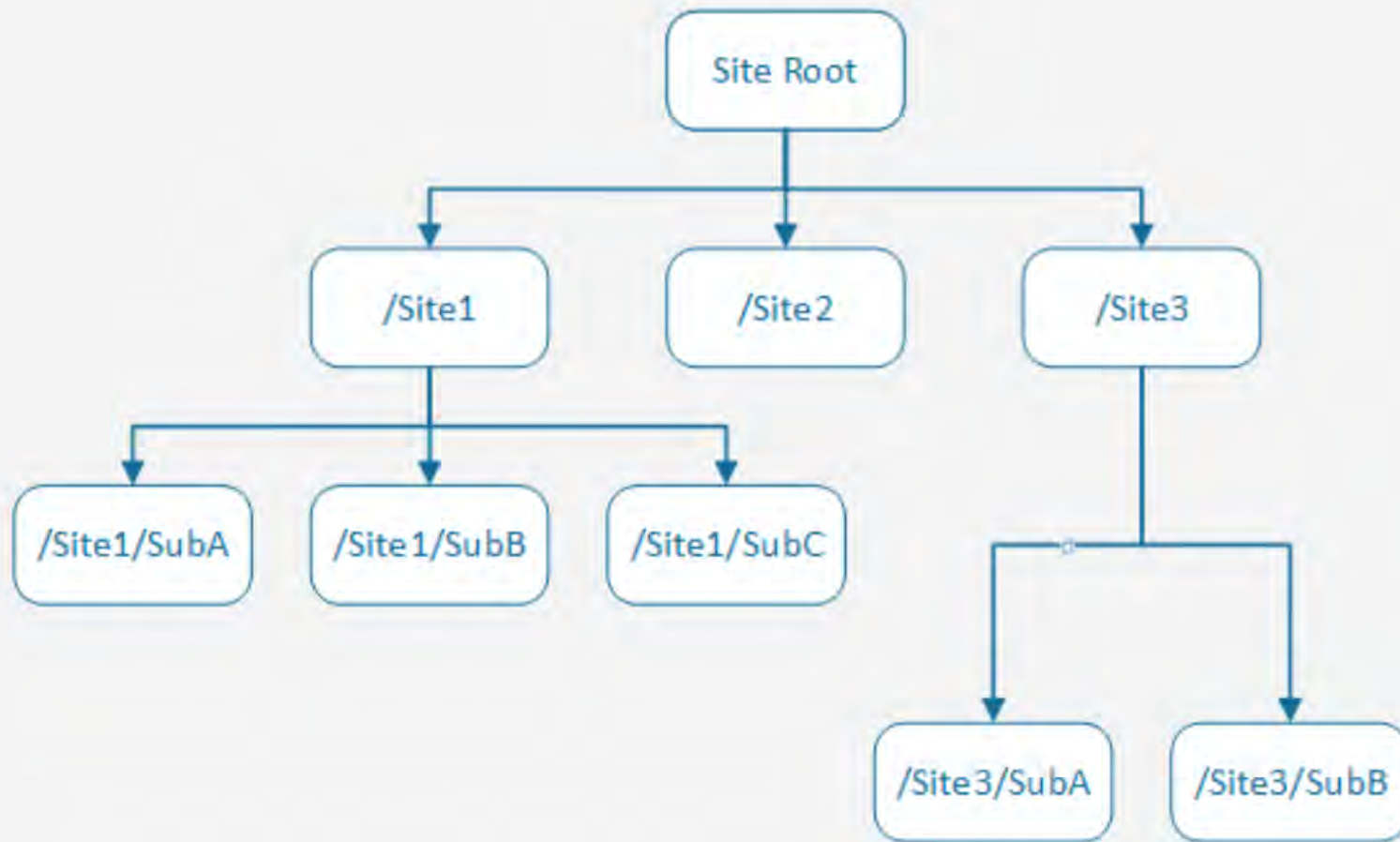
- \* building each site node generates roughly 8 SQL Server round trips
- \* These sitemaps are then stored in the Object Cache on WFEs

Navigation style  
has a huge impact  
on performance.



8 site nodes/~64 SQL calls





8 site nodes/~64 SQL calls



## Better Options for Navigation

- Managed Navigation (i.e., using a term set to drive navigational structures) can significantly improve page performance.  
*note: the SharePoint Server Publishing Infrastructure site collection Feature must be enabled to use Managed Navigation*
- Search-driven navigation leverages SharePoint's Search index and the process of client-side navigational rendering to dramatically speed things up.  
*note: implementation is non-trivial and less customizable*

Using structural navigation is the default (but poor) choice for complex site hierarchies in the cloud.

- \* building each site node generates roughly 8 SQL Server round trips
- \* These sitemaps are then stored in the Object Cache on WFEs

## Navigation style



As was pointed-out in the navigational scenario,  
Search can be used to boost performance significantly.

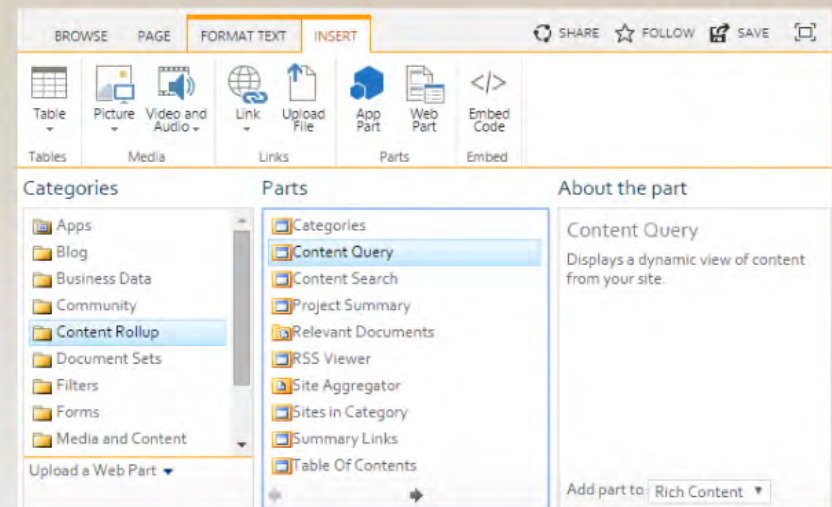


In the cloud, the CQWP can cause some signifi

ut in the navigational scenario,  
ed to boost performance significantly.



Do you like the Content  
Query Web Part (CQWP)?

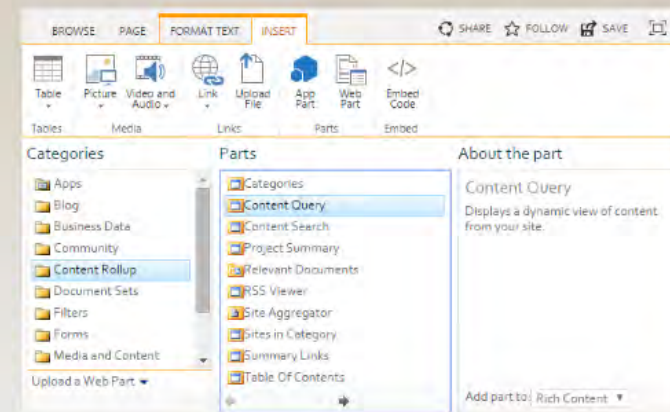


the CQWP can cause some significant performance hits.

As was pointed-out in the navigational scenario,  
Search can be used to boost performance significantly.

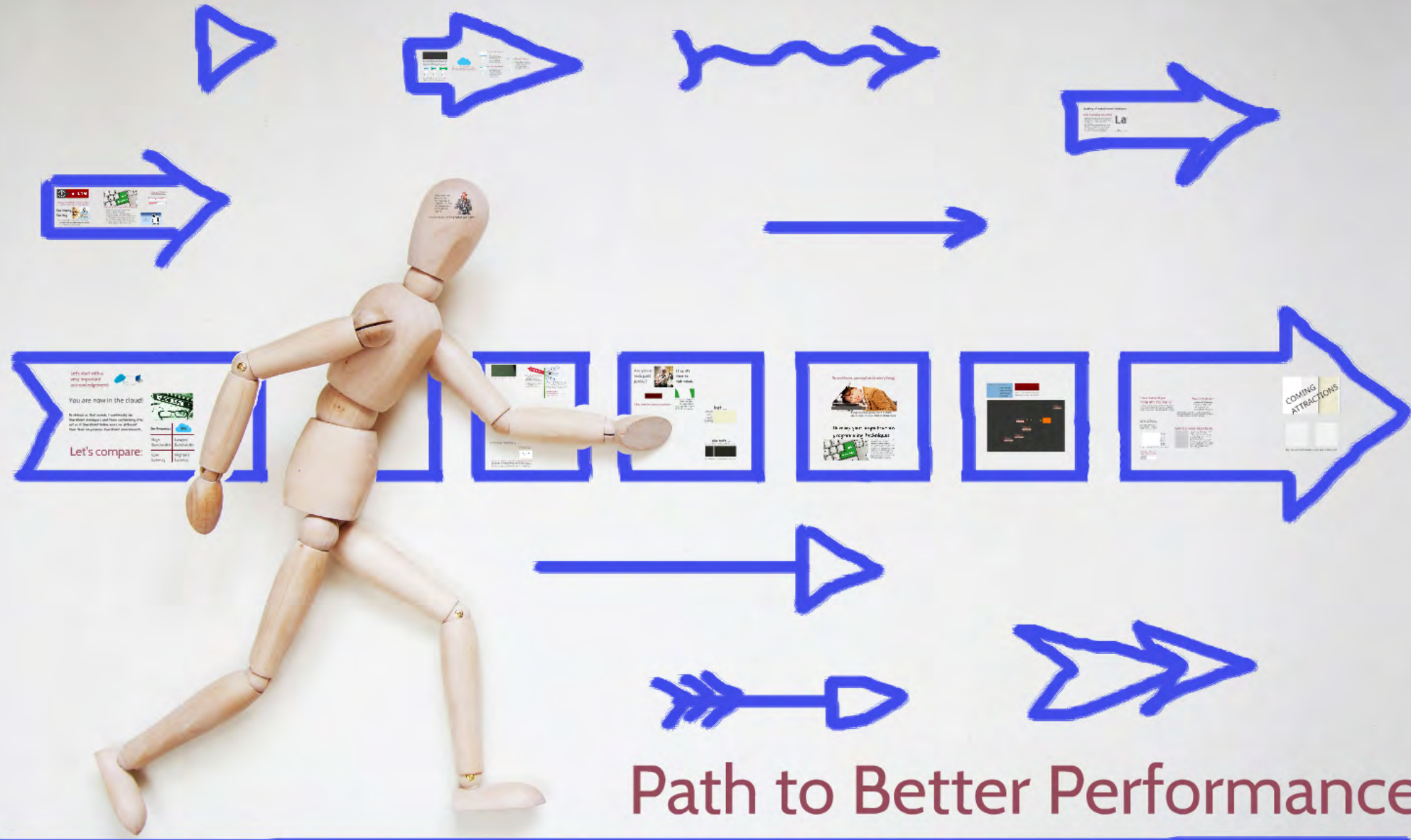


Do you like the Content  
Query Web Part (CQWP)?



**In the cloud, the CQWP can cause some significant performance hits.**

- The CQWP performs expensive cross-list and cross-site queries at run-time.
- The CQWP relies on the Object Cache to store results for acceptable performance.
- The Content Search Web Part (CSWP) provides options that are similar to the CQWP (and in a number of ways, more powerful) and uses Search so it's FAST!

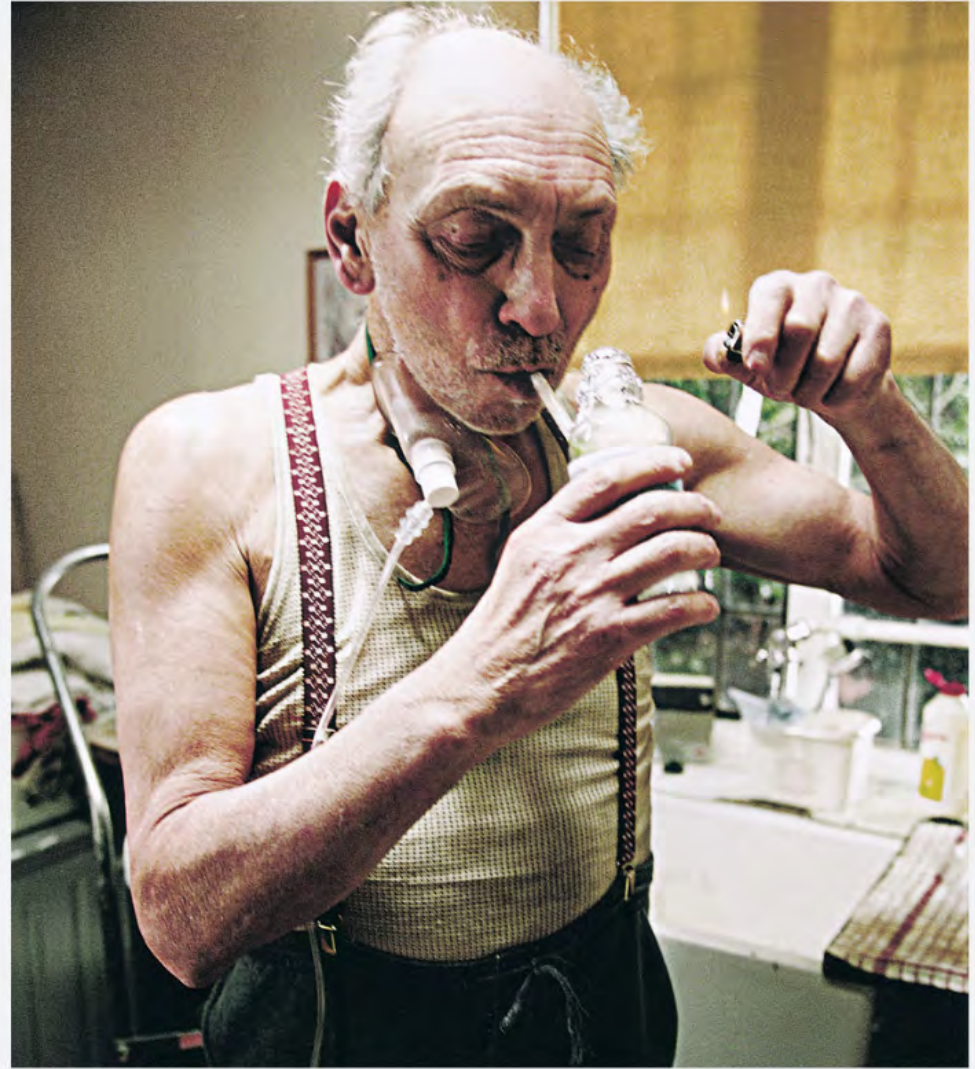


Path to Better Performance



Okay, time for a serious question ...

Are you a  
web part  
junkie?

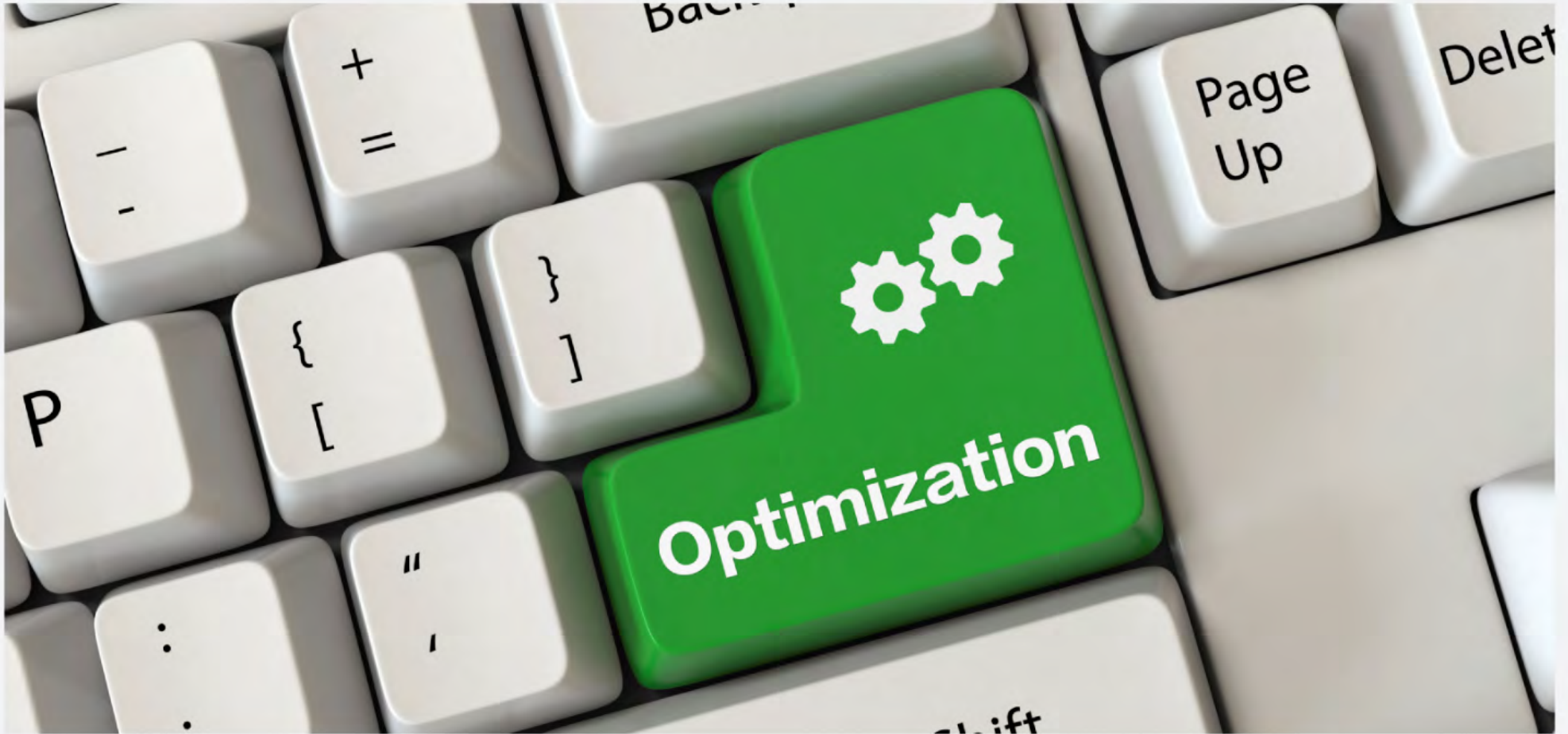






If so, it's  
time to  
talk rehab.

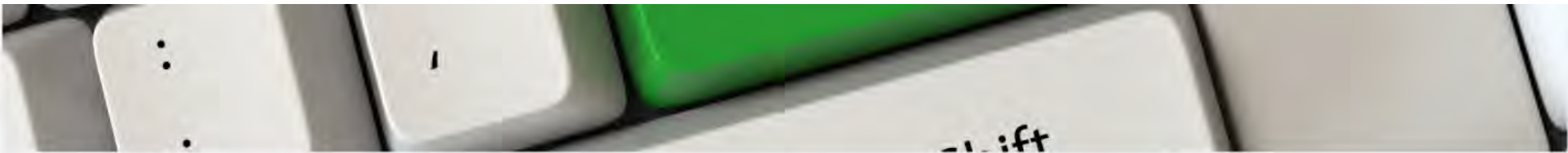




What's the alternative to freebasing web parts?



**There's no single**



What's the alternative to freebasing web parts?



**There's no single  
(or simple) answer.**

Generally speaking, consider leveraging client-side code (JavaScript) and asynchronous techniques - both of which we'll discuss soon.

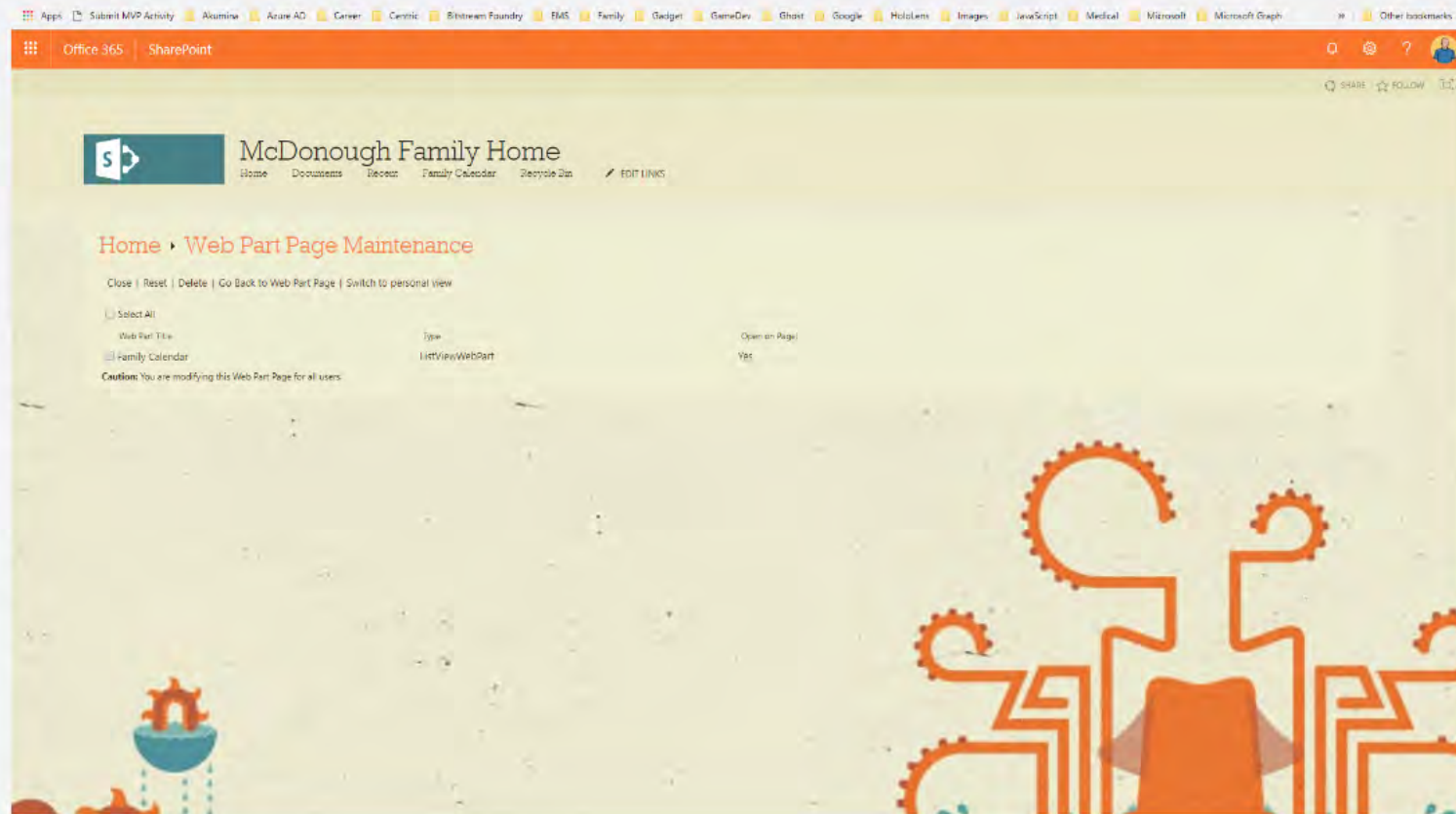
but ...

Be sure to put  
problem pages

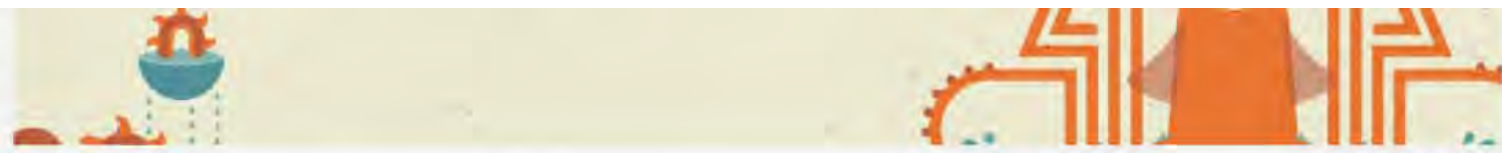


# but ...

Be sure to put problem pages in web part maintenance view with **?contents=1** to find web parts which are closed but not deleted!



which are closed  
but not deleted!



**also note ...**

When I say "web part," I'm talking about traditional (server-side) web parts. Everything is different, and all bets are off with SPFx/client-side web parts.

# also note ...

When I say "web part," I'm talking about traditional (server-side) web parts. Everything is different, and all bets are off with SPFx/client-side web parts.

```
HELLOWORD-WEBPART
├── .vscode
├── config
├── dist
├── lib
├── node_modules
├── src
├── temp
├── .editorconfig
├── .gitignore
├── .yo-rc.json
├── gulpfile.js
├── package-lock.json
├── package.json
├── README.md
├── tsconfig.json
└── tslint.json
```

```
yo @microsoft/sharepoint

Welcome to the
SharePoint Client-side
Solution Generator

Let's create a new SharePoint solution.
? What is your solution name? helloworld-webpart
? Which baseline packages do you want to target for your component(s)? SharePoint Online only (latest)
? Where do you want to place the files? Use the current folder
? Do you want to allow the tenant admin the choice of being able to deploy the solution to all sites immediately
ny feature deployment or adding apps in sites? No
? Which type of client-side component to create? WebPart
? What is your Web part name? HelloWorld
? What is your Web part description? HelloWorld description
? Which framework would you like to use? (Use arrow keys)
> No JavaScript framework
  React
  Knockout
```

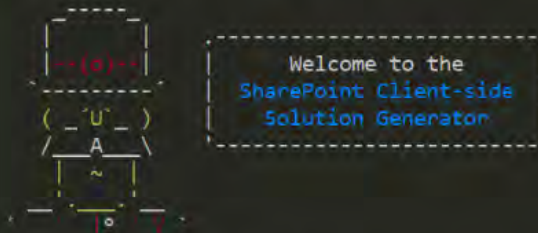
And a friendly reminder: use a CDN with those SPFx web parts!

# also note ...

When I say "web part," I'm talking about traditional (server-side) web parts. Everything is different, and all bets are off with SPFx/client-side web parts.

```
HELLOWORD-WEBPART
├── .vscode
├── config
├── dist
├── lib
├── node_modules
├── src
├── temp
├── .editorconfig
├── .gitignore
├── .yo-rc.json
├── gulpfile.js
├── package-lock.json
├── package.json
├── README.md
├── tsconfig.json
└── tslint.json
```

```
yo @microsoft/sharepoint
```

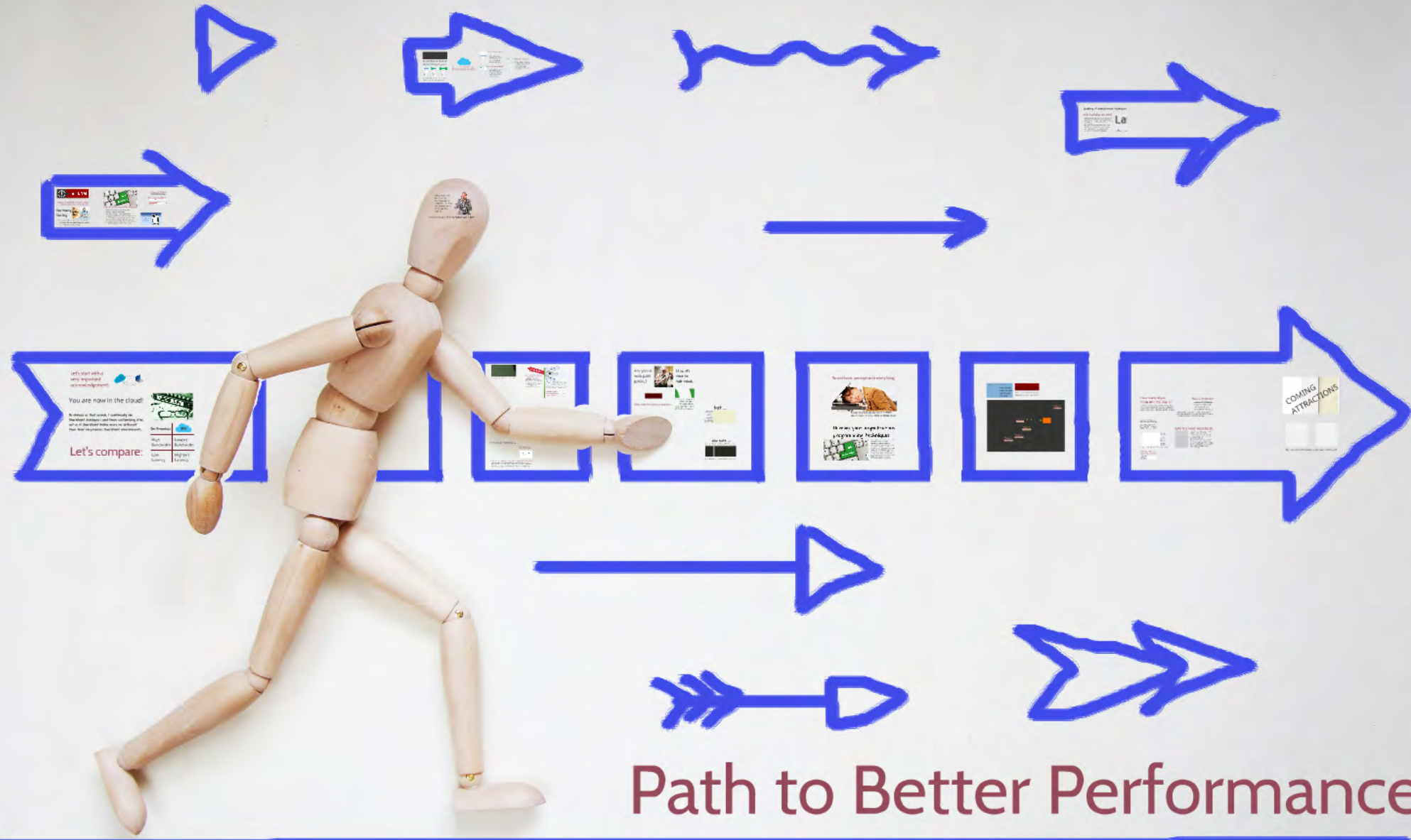


```
Let's create a new SharePoint solution.
```

```
? What is your solution name? helloworld-webpart
? Which baseline packages do you want to target for your component(s)? SharePoint Online only (latest)
? Where do you want to place the files? Use the current folder
? Do you want to allow the tenant admin the choice of being able to deploy the solution to all sites immediately by feature deployment or adding apps in sites? No
? Which type of client-side component to create? WebPart
? What is your Web part name? HelloWorld
? What is your Web part description? HelloWorld description
? Which framework would you like to use? (Use arrow keys)
> No JavaScript framework
  React
  Knockout
```

And a friendly reminder: use a CDN with those SPFx web parts!





Path to Better Performance



Sometimes, perception is everything.



**ALARM**

A page may load quickly, but if it FEELS slow to users, it is the SAME AS BEING SLOW.

# Sometimes, perception is everything.



**ALARM**

A page may load quickly, but if it **FEELS** slow to users, it is the **SAME AS BEING SLOW.**

# Develop your asynchronous programming techniques



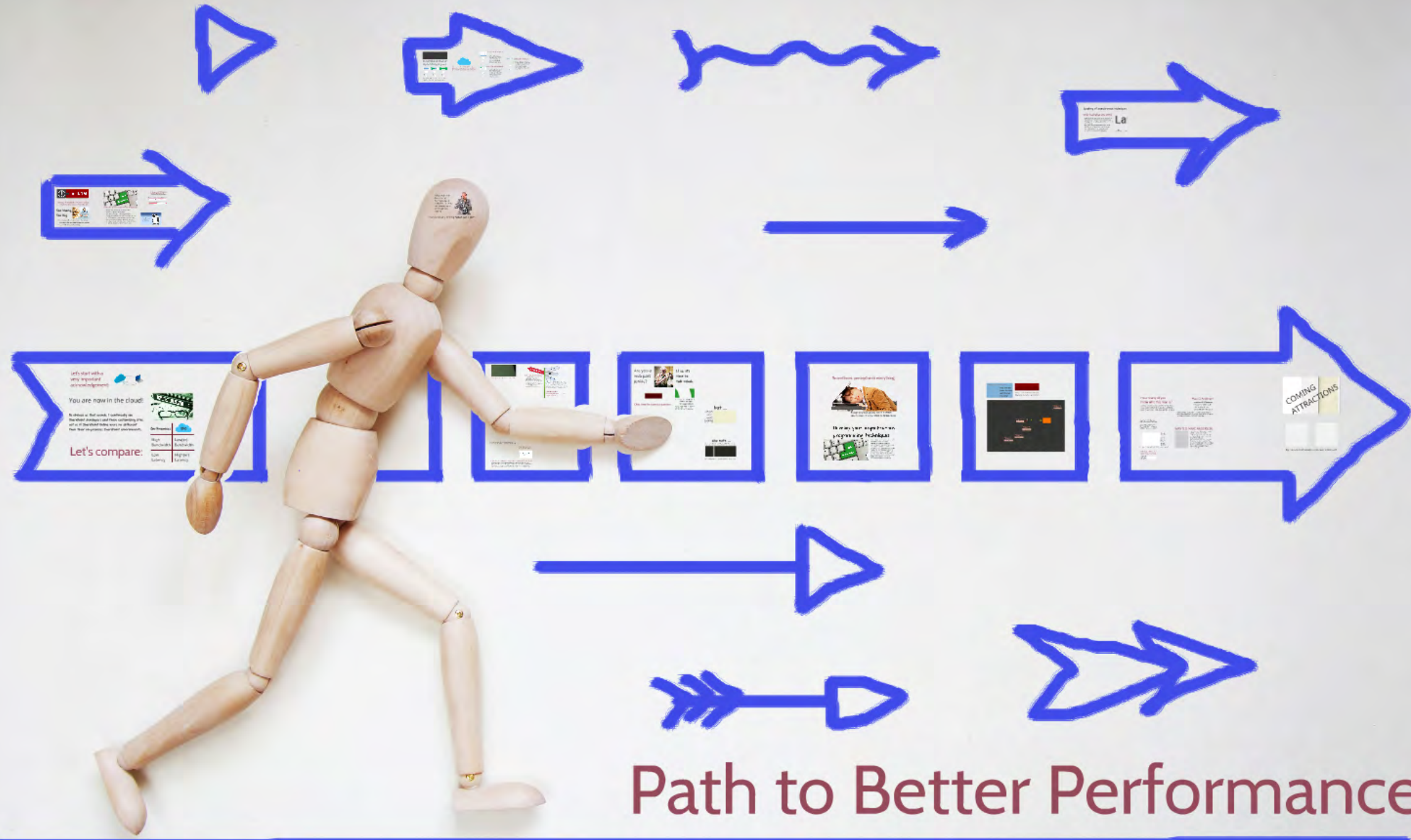
- You can't block a browser's main thread of execution, so leveraging async development patterns is essential.
- Async programming is made much easier in jQuery using promises. Promises approximate a synchronous programming model under asynchronous conditions.
- Certain web parts (e.g., the CSWP) also allow you to set their (a)sync behavior.
- Good use of async techniques make pages **appear** to load faster ... and as we discussed, perception is everything.

# Speaking of asynchronous techniques:

## Only load what you need.

- Instead of fetching everything at once within the context of the initial page load, retrieve the page with only the payload that's needed immediately.
- (Lazy) load images and other items "below the fold" only if users start scrolling down and will see them (e.g., Facebook and LinkedIn's "forever-scrolling" pages).





# Path to Better Performance

Ask yourself this question:



How well do I  
know the code  
and libraries  
I'm using?

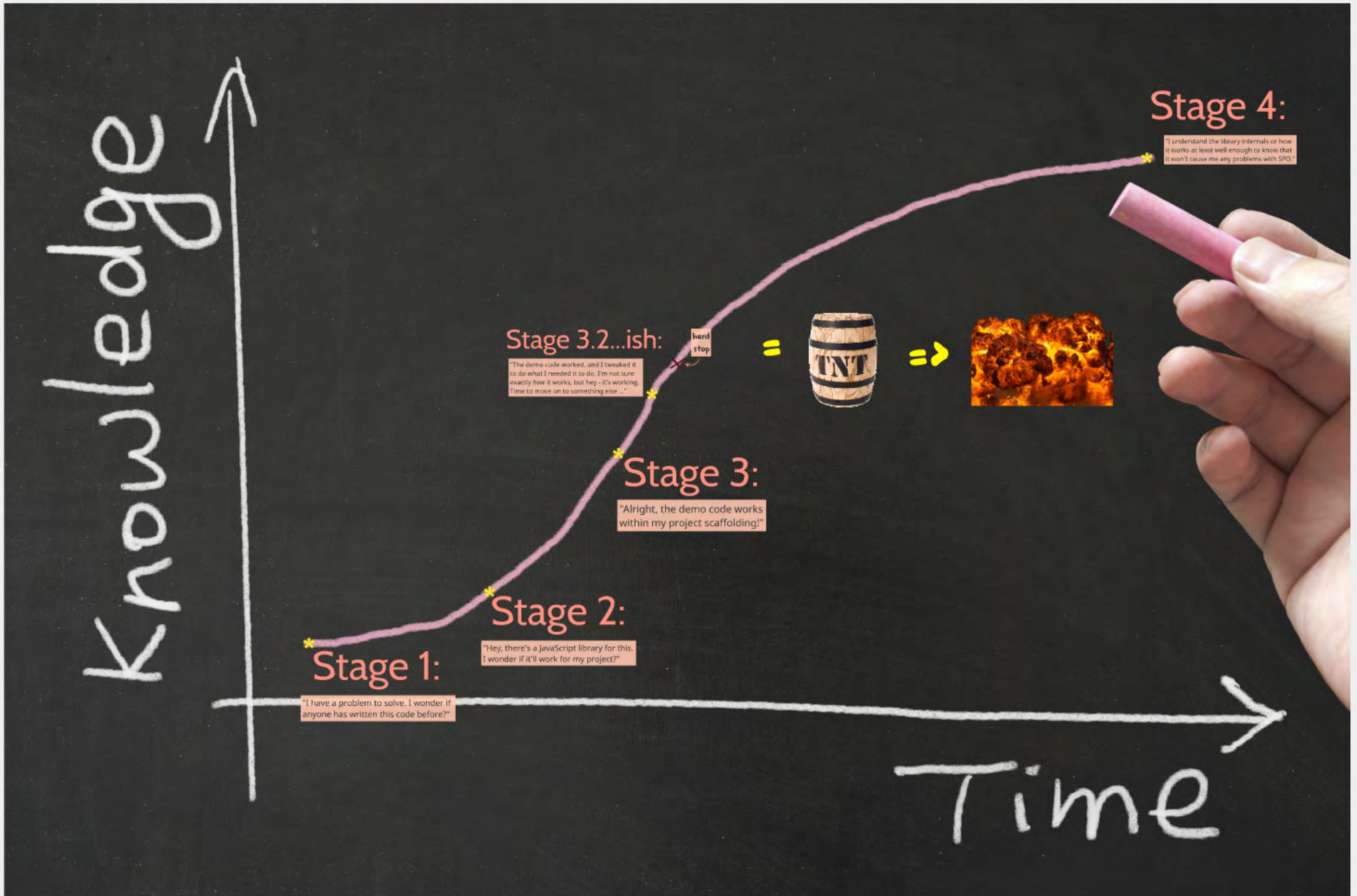




Reason I ask: when troubleshooting performance issues, I commonly encounter a pattern that can be illustrated with the following diagram of stages:

I'm using?

be illustrated with the following diagram of stages:



"Alright,  
within m



\*  
Stage 1:

"I have a problem to solve. I wonder if anyone has written this code before?"

\*  
Stage 2:

"Hey, there's a JavaScript library for this. I wonder if it'll work for my project?"



# Stage 3:

"Alright, the demo code works within my project scaffolding!"

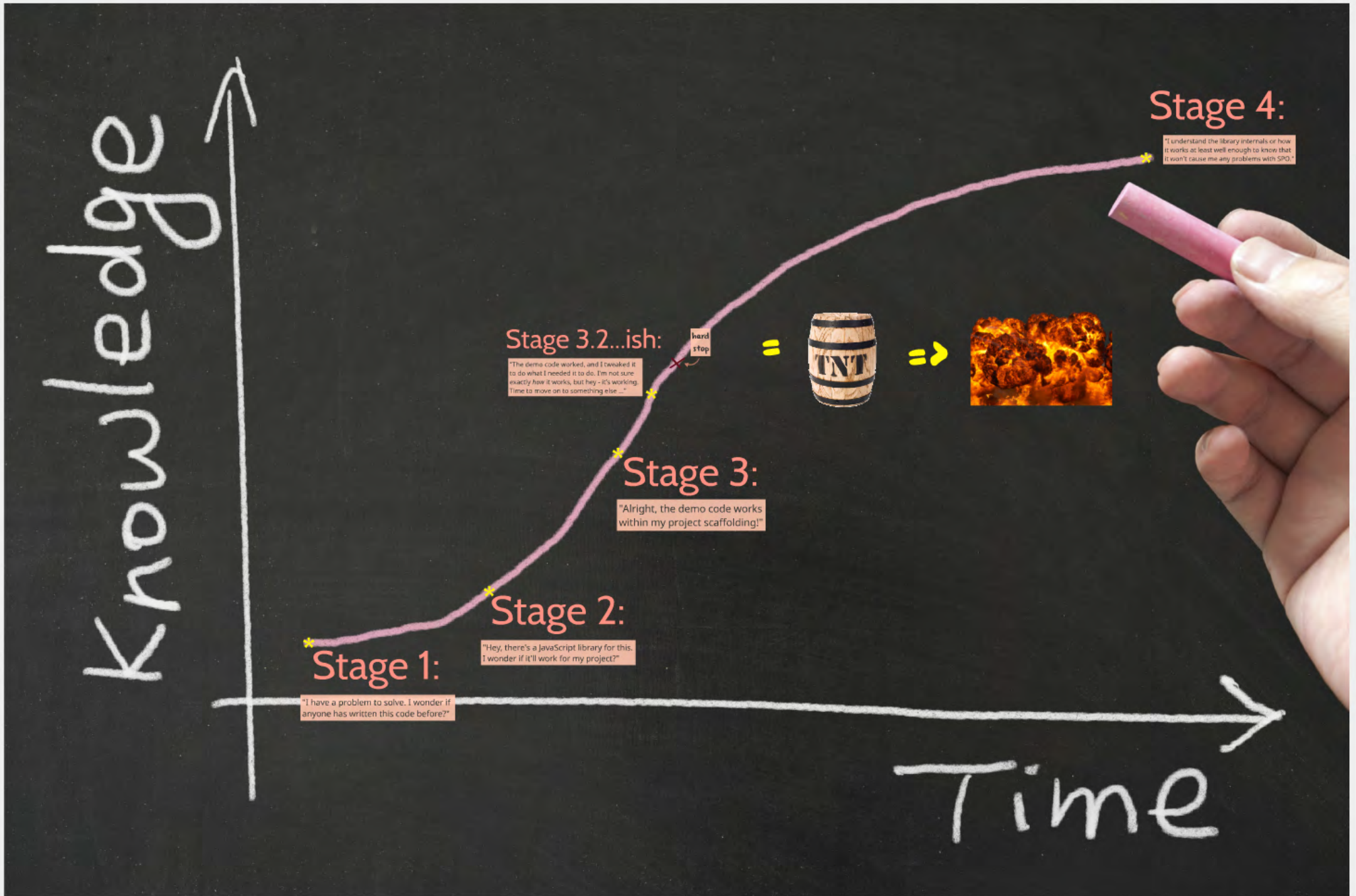
# Stage 4:

"I understand the library internals or how it works at least well enough to know that it won't cause me any problems with SPO."



I'm using?

be illustrated with the following diagram of stages:



# Stage 3.2...ish:

"The demo code worked, and I tweaked it to do what I needed it to do. I'm not sure exactly *how* it works, but hey - it's working. Time to move on to something else ..."

hard  
stop

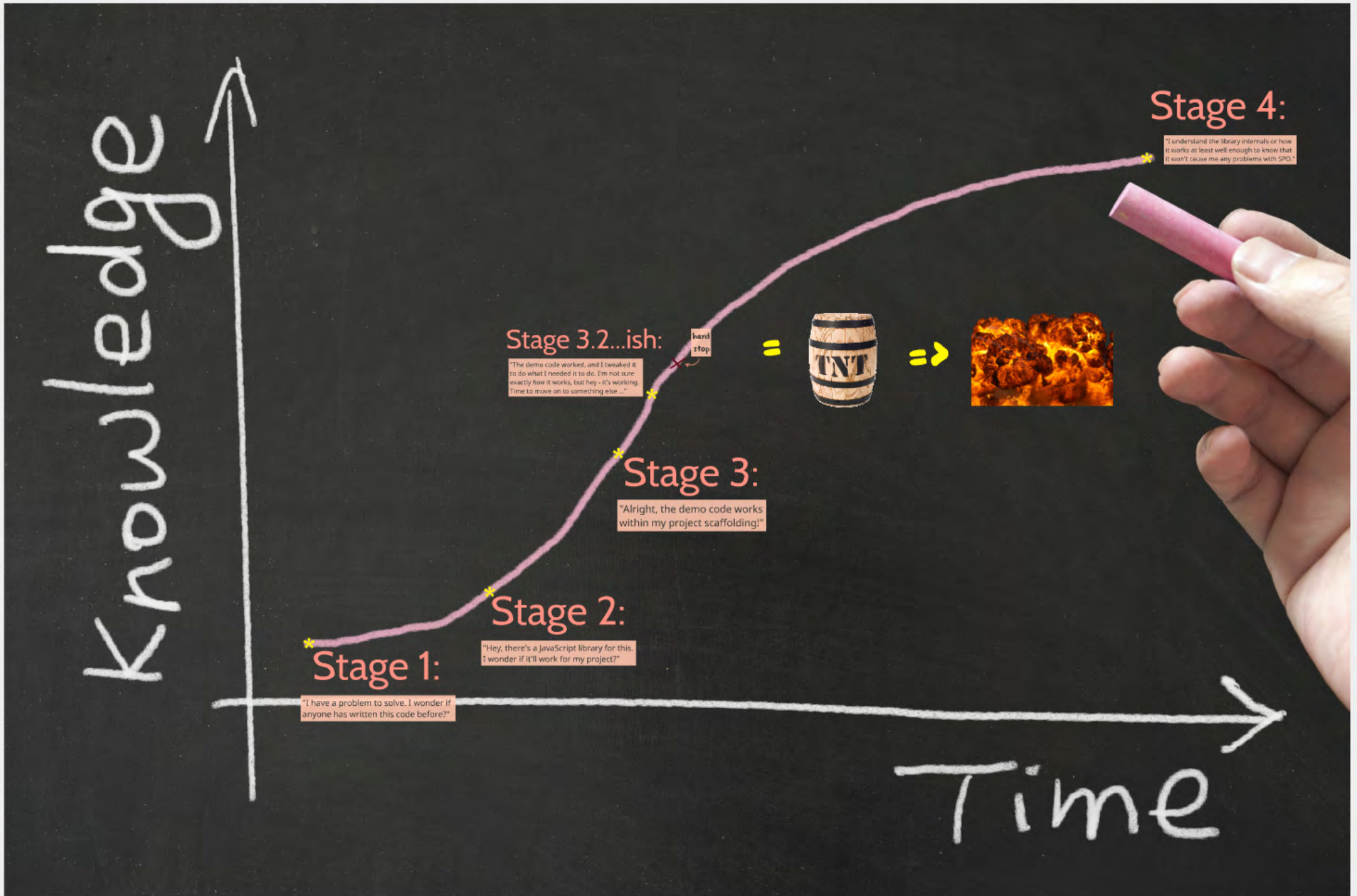


# Stage 3:

What is the next stage?

I'm using?

be illustrated with the following diagram of stages:





example

How many of you



example

## How many of you know who this man is?

Chances are at least one or two of you have used code that he has created to get things done in your client-side development projects ...



Before CSOM/JSOM and REST  
Marc's library simplified access  
for developers everywhere. It



# Marc D. Anderson

- **creator of SPServices**
- master of client-side development and associated techniques

Before CSOM/JSOM and REST APIs - and before WCF SVC endpoints - Marc's library simplified access to the older ASMX web service endpoints for developers everywhere. It's still used heavily today.

So, getting back to "know your code/libraries" and how they work ...

As Marc will tell you, SPServices works just

So, getting back to "know your code/libraries" and how they work ...

As Marc will tell you, SPServices works just fine with SharePoint Online. But even Marc will tell you that you probably shouldn't use all of SPServices' methods when accessing SPO.

```
1 //Pre-populate all "Contact" fields with current user
2 var thisUserName = $().SPServices.SPGetCurrentUser({
3     fieldName: "Title",
4     debug: false
5 });
6 $().SPServices.SPFindPeoplePicker({
7     peoplePickerDisplayName: "Contact",
8     valueToSet: thisUserName,
9     checkNames: true
10 });
11 $().SPServices.SPFindPeoplePicker({
12     peoplePickerDisplayName: "Author/Contact",
13     valueToSet: thisUserName,
14     checkNames: true
15 });
16 $().SPServices.SPFindPeoplePicker({
17     peoplePickerDisplayName: "Organizer/Contact",
18     valueToSet: thisUserName,
19     checkNames: true
20 });
```

Consider  
this code.

It works just  
fine and does  
exactly what  
the comment  
indicates.

**But it has a  
big problem.**

Anyone ever used the SPServices.SPGetCurrentUser() method

```
1 //Pre-populate all "Contact" fields with current user
2 var thisUserName = $().SPServices.SPGetCurrentUser({
3     fieldName: "Title",
4     debug: false
5 });
6 $().SPServices.SPFindPeoplePicker({
7     peoplePickerDisplayName: "Contact",
8     valueToSet: thisUserName,
9     checkNames: true
10 });
11 $().SPServices.SPFindPeoplePicker({
12     peoplePickerDisplayName: "Author/Contact",
13     valueToSet: thisUserName,
14     checkNames: true
15 });
16 $().SPServices.SPFindPeoplePicker({
17     peoplePickerDisplayName: "Organizer/Contact",
18     valueToSet: thisUserName,
19     checkNames: true
20 });
```

Consider  
this code.

It works just  
fine and does  
exactly what  
the comment  
indicates.

**But it has a  
big problem.**

Has anyone ever used the SPServices.SPGetCurrentUser() method?

Switching over to REST-based calls

# WANTED: MARC ANDERSON



- Under the hood, SPGetCurrentUser() is generating an additional call to `/_layouts/userdisp.aspx` to “scrape” the contents of the page that is returned.
- If you (innocently) use SPGetCurrentUser() in your JavaScript files (especially multiple times in the context of a single page), you're creating all sorts of additional load on SPO and delaying the final results of your executing scripts.

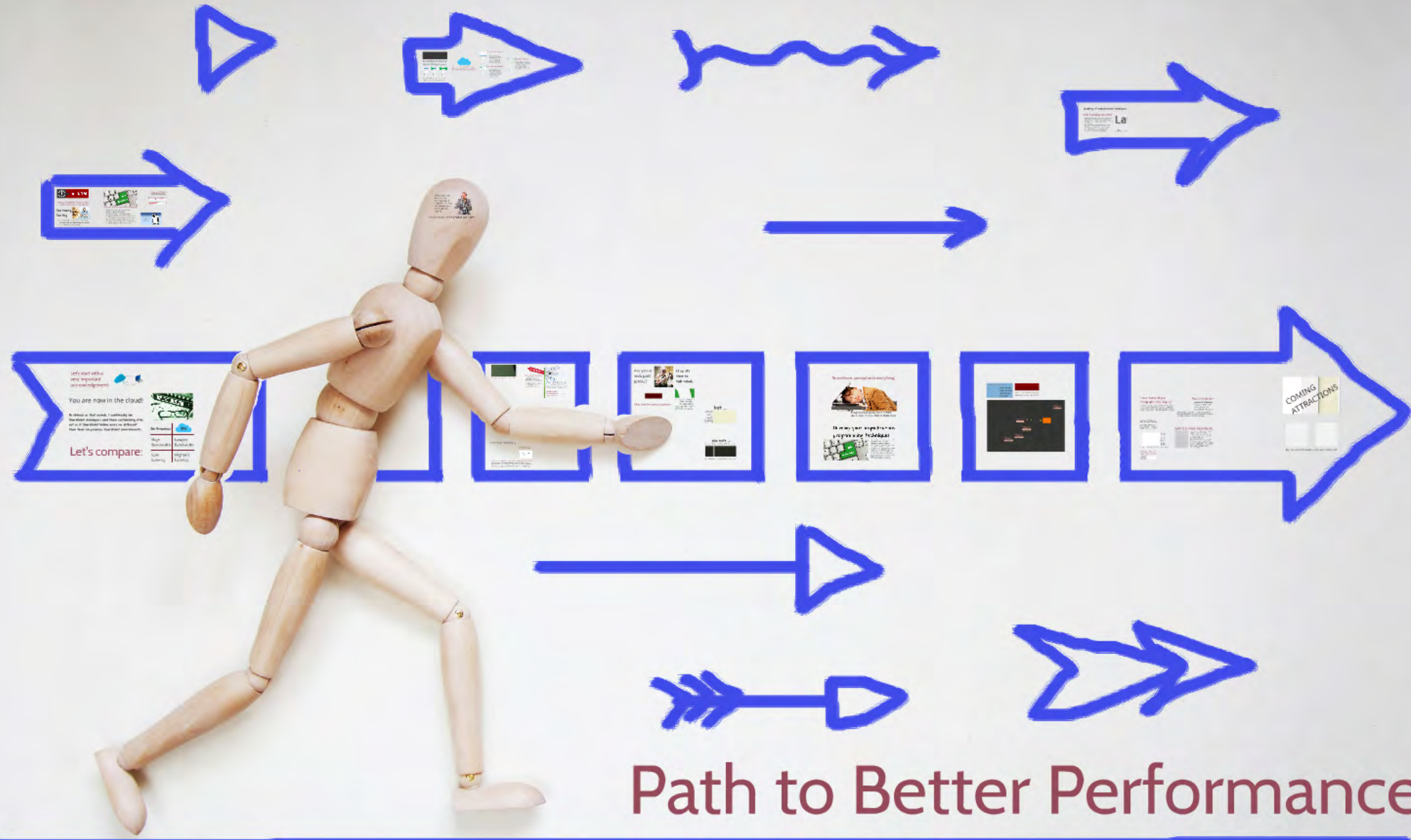


# Switching over to REST-based calls to get current user information can dramatically reduce execution time.

We had a script where `SPGetCurrentUser()` was being called several times. The results from swapping-in REST-based calls for the `SPGetCurrentUser()` calls:

I also performed some basic tests to capture the speed differences. I performed each test 10 times and here are the results:

- \* Without the fix or browser caching - avg. 14.47 seconds
- \* With the fix without browser caching - avg. 7.17 seconds
- \* With the fix and browser caching - avg. 5.84 seconds



# Path to Better Performance



### Page diagnostics for SharePoint

About

CorrelationId: 4505889e-e08f-6000-387b-395bcc7b5bff  
SPRequestDuration: N/A  
SPIISLatency: N/A  
Page load time: 984ms  
Url: https://mcdonoughfamilyonline.sharepoint.com/SitePages/Home.aspx

Stop

Diagnostic Network trace

- Check Running as Standard User
- Check Requests To SharePoint
- Check using CDN
- Check for Large Image Sizes

### Page diagnostics for SharePoint

About

CorrelationId: 4505889e-e08f-6000-387b-395bcc7b5bff  
SPRequestDuration: N/A  
SPIISLatency: N/A  
Page load time: 984ms  
Url: https://mcdonoughfamilyonline.sharepoint.com/SitePages/Home.aspx

Stop

Diagnostic Network trace

Export to JSON

- .../WsaUpload.ashx 77.50ms
- .../Home.aspx 984.00ms
- .../ 31.30ms
- .../oslo.css 5.50ms

Big thanks to Scott Stewart and his team at Microsoft!



# The Quick Summary



- Don't treat SPO like your on-premises SharePoint farm. The two operate differently.
- Server-based caching isn't your friend (generally speaking) in SPO.
- Your browser can be your best friend when trying to troubleshoot SPO performance issues.
- Know the code you implement - or at least profile it before release.

# References

SPTechCon Austin 2016

<http://www.sptechcon.com/>

“Technical diagrams for SharePoint 2013”

<https://technet.microsoft.com/en-us/library/cc263199.aspx>

“Office 365 Performance Management”

<https://mva.microsoft.com/en-US/training-courses/office-365-performance-management-8416>

# References

“Where is my data?”

[https://www.microsoft.com/online/legal/v2/en-us/MOS\\_PTC\\_Geo\\_Boundaries.htm](https://www.microsoft.com/online/legal/v2/en-us/MOS_PTC_Geo_Boundaries.htm)

“Deploying Office 365”

[http://video.ch9.ms/sessions/project/2014/PC215\\_Medford.pptx](http://video.ch9.ms/sessions/project/2014/PC215_Medford.pptx)

“New datacenter regions for Office 365”

<https://support.office.com/en-us/article/Network-and-migration-planning-for-Office-365-f5ee6c33-bcd7-4b0b-b0f8-dc1d9fb8d132?ui=en-US&rs=en-US&ad=US#calculators>



# References

“Network and migration planning for Office 365”

<https://support.office.com/en-us/article/Network-and-migration-planning-for-Office-365-f5ee6c33-bcd7-4b0b-b0f8-dc1d9fb8d132?ui=en-US&rs=en-US&ad=US#calculators>

Fiddler web debugging proxy

<http://www.telerik.com/fiddler>

“FAQ – Certificates in Fiddler”

<http://www.telerik.com/blogs/faq---certificates-in-fiddler>

# References

“Analyzing your webpage’s network traffic”

[https://msdn.microsoft.com/en-us/library/dn255004\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/dn255004(v=vs.85).aspx)

“Diagnosing performance issues with SharePoint Online”

<https://support.office.com/en-us/article/Diagnosing-performance-issues-with-SharePoint-Online-3c364f9e-b9f6-4da4-a792-c8e8c8cd2e86>

“Image optimization for SharePoint Online”

<https://support.office.com/en-gb/article/Image-optimization-for-SharePoint-Online-c7edb02a-fdab-4f91-9a20-cba01dad28ef>

# References

Font Awesome

<https://fontawesome.github.io/Font-Awesome/>

“Use the Office 365 content delivery network with SharePoint Online”

<https://docs.microsoft.com/en-us/office365/enterprise/use-office-365-cdn-with-spo>

“Navigation options for SharePoint Online”

<https://support.office.com/en-us/article/Navigation-options-for-SharePoint-Online-adb92b80-b342-4ecb-99a1-da2a2b4782eb>

# References

“Using Content Search Web Part instead of Content Query Web Part to improve performance in SharePoint Online”

<https://support.office.com/en-us/article/Using-Content-Search-Web-Part-instead-of-Content-Query-Web-Part-to-improve-performance-in-SharePoint-Online-e8ce6b72-745b-464a-85c7-cbf6eb53391b>

“Using jQuery Promises & Deferreds with SharePoint 2013”

<http://blog.qumsieh.ca/2013/10/31/using-jquery-promises-deferreds-with-sharepoint-2013-jsom/>

# References

“Lazy Load Plugin for jQuery”

[https://github.com/tuupola/jquery\\_lazyload](https://github.com/tuupola/jquery_lazyload)

“Documentation > \$.SPServices.SPGetCurrentUser”

<http://spservices.codeplex.com/wikipage?title=%24%28%29.SPServices.SPGetCurrentUser>

“SPServices Example: UserProfileService.GetUserProfileByName”

<http://sympmarc.com/2012/02/15/spservices-example-userprofileservice-getuserprofilebyname/>

# References

"get current user info using jquery, REST and csom"

<http://sharepoint.stackexchange.com/questions/124909/get-current-user-info-using-jquery-rest-and-csom>

"Users, groups and the roles REST API reference"

[https://msdn.microsoft.com/en-us/library/office/dn531432.aspx#bk\\_User](https://msdn.microsoft.com/en-us/library/office/dn531432.aspx#bk_User)

"Caching, You Ain't No Friend Of Mine"

<http://sharepointinterface.com/2016/01/31/caching-you-aint-no-friend-of-mine/>

# References

“Deprecation of Custom Code in Sandboxed Solutions”

<https://blogs.msdn.microsoft.com/sharepointdev/2014/01/14/deprecation-of-custom-code-in-sandboxed-solutions/>

“Using the Object Cache with SharePoint Online”

<https://support.office.com/en-us/article/Using-the-object-cache-with-SharePoint-Online-38bc9c14-3826-449c-beb6-b1003bcbeaaf>

# References

"Tune SharePoint Online performance"

<https://aka.ms/spoperformance>

"Use the Page Diagnostics tool for SharePoint Online"

<https://docs.microsoft.com/en-us/office365/enterprise/page-diagnostics-for-spo>

"Microsoft 365 Engineering Webcast Series: Optimizing Classic pages in SharePoint Online"

<https://microsoft365.eventbuilder.com/event?eventid=u4n0g7>



Thank you

# Session Feedback

Please complete session feedback form located in your attendee bag and return to the registration table once completed.



# Thank you to our Sponsors

## Platinum Sponsors



## Gold Sponsors



## Raffle Sponsors



# SHAREPINT

SPONSORED BY:



## The Summit Restaurant

- *Located inside Cincinnati State Technical College*

**Time: 5:30pm to ???**

Remember to get your complimentary drink tickets during Closing Remarks!





# Sean P. McDonough

SharePoint & Office 365

Gearhead, Tinkerer, and

Microsoft MVP

**Email:** [sean@bitstreamfoundry.com](mailto:sean@bitstreamfoundry.com)

**Twitter:** [@spmcdonough](https://twitter.com/spmcdonough)

**Blog:** <http://SharePointInterface.com>

**About:** <http://about.me/spmcdonough>