



Zowe Joint Community 20PI4 (Q4) Context & Vision

Presenters:

Michael DuBois, Bruce Armstrong, Peter Fandel
Mark Ackert, Joe Winchester, Sean Grady

Agenda

- Zowe Deliverables and achievements from last PI
- Community highlights from last PI
- New Technical Steering Committee – why and what and who?
- OMP Mentorship – Zebra
- ZLC Review/Comments on Squads High Level Plans

What did we deliver last PI?

20PI3 Zowe Achievements



Zowe CLI Squad

- ❖ Zowe Client Python SDK is published to the Python Package Index
- ❖ Zowe Client Swift SDK is available in GitHub
- ❖ Zowe Client Node SDK is nearly published to public npm

Zowe API Squad

- ❖ Removed the dependency on z/OSMF for authentication by adding SAF as an additional authentication provider for tokens
- ❖ Swagger URL is now provided for z/OSMF
- ❖ The default configuration now supports character encoding
- ❖ Welcomed two new members, Jordan Cain and Carson Cook, to the squad

Zowe App Framework Squad

- ❖ Developed an improved plan for HA/FT which significantly reduces complexity while improving user experience
- ❖ Created a Dockerized release of Zowe's server runtimes as an option to run under Linux as an alternative to z/OS
- ❖ Ran a mentorship program with OMP for 4 mentees to enhance the App Framework and kickstart new app development

Zowe Explorer Squad

- ❖ Introduced a configurable / toggleable profile validation mechanism
- ❖ Improved installation speed and overall performance by bundling using Webpack
- ❖ "Allocate Like" for data set allocation
- ❖ Grouped favorites into a more consistent profile-specific view
- ❖ Finished refactoring unit & integration tests

Zowe Documentation Squad

- ❖ API Reference Guide is now available
- ❖ Automated the Release Notes process
- ❖ Documented the Zowe Client SDKs
- ❖ Documented how to leverage the SSO / MFA features in Zowe CLI using the new "base profile" to store a token

Zowe Onboarding Squad

- ❖ Completed V1 Conformance Program including Test Criteria Change Process
- ❖ New vendor (SEGUS) has earned a Zowe V1 Conformance badge for App Framework
- ❖ Established a new Quarterly Webinar Series
- ❖ Played a major role in transitioning the courseware for Master the Mainframe to Zowe Explorer

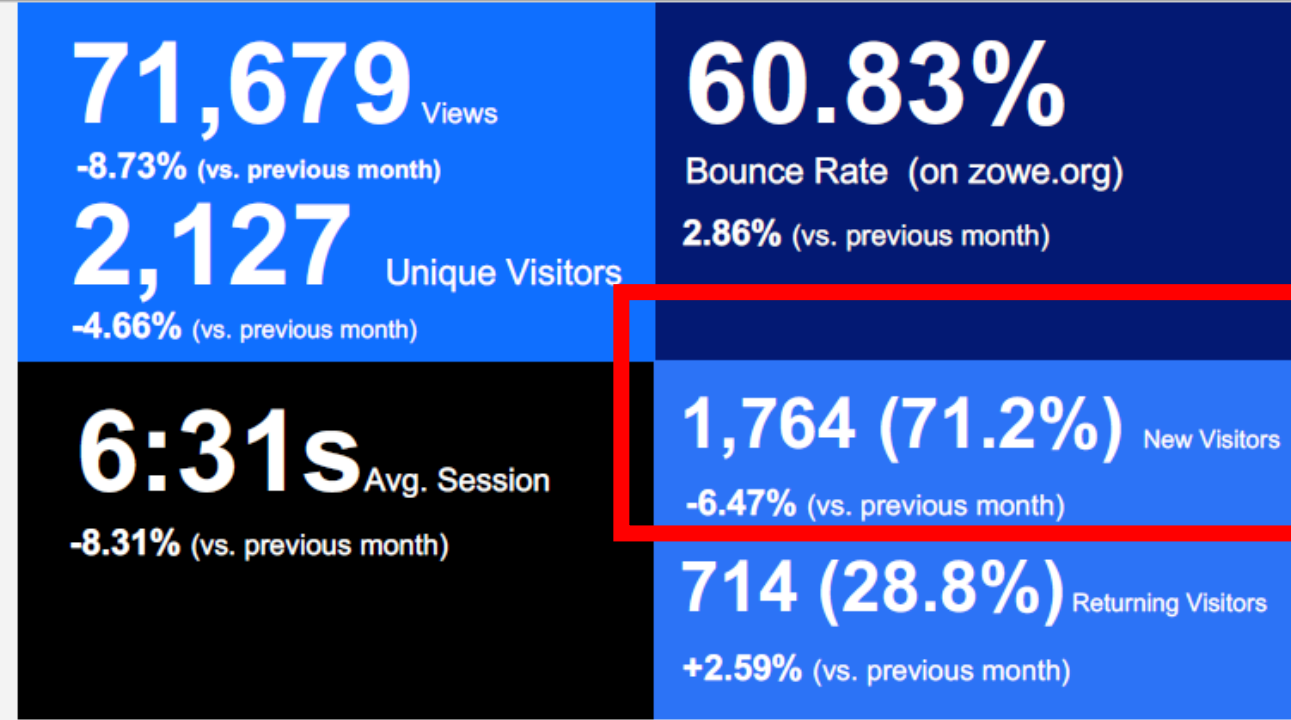
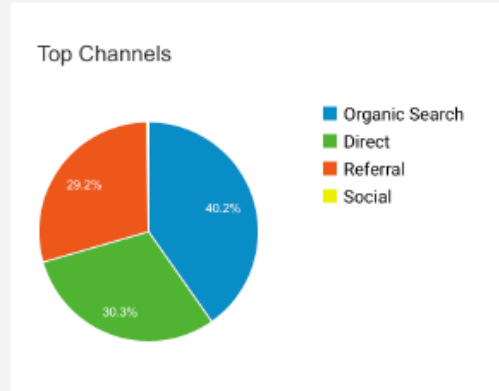
Community Highlights

Zowe.org

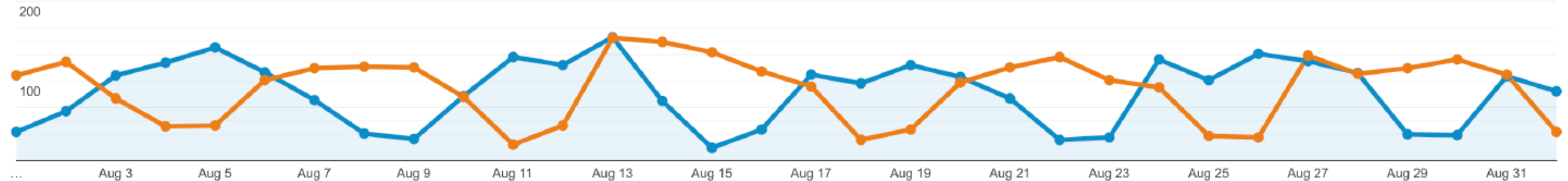
- 1,612,960 Cumulative Pageviews since Launch (Aug 13th, 2018)
- 21.7% of visitors go to the download page from homepage in 1st interaction
 - 17% in June 2020
- Avg 11.02 Pages/Session (-0.92% vs July 2020)









Top Referral Sites:

- Zowe.org: 39.70% (-18.54%)
- Openmainframeproject.org: 21.48% (-6.45%)
- IBM.com: 6.81% (43.75%)
- Github.com: 4.15% (-3.45%)



Aug 1, 2020 - Sep 1, 2020: ● Users
Jul 1, 2020 - Aug 1, 2020: ● Users



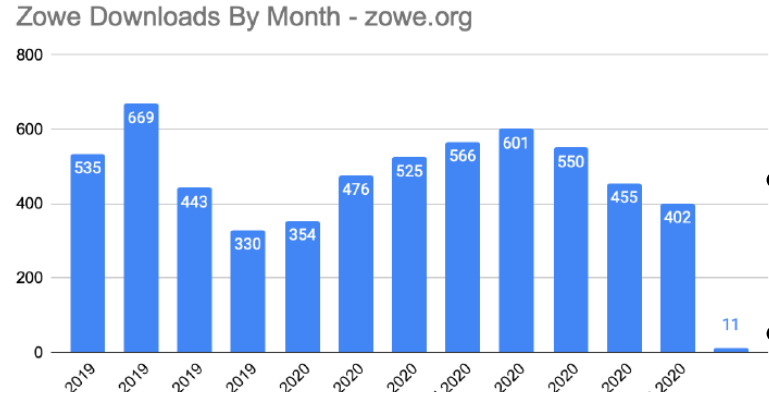
Country [?]	Acquisition	
	Users [?] ↓	New Users [?]
	2,060 % of Total: 100.00% (2,060)	1,708 % of Total: 100.12% (1,706)
1.  United States	645 (31.02%)	531 (31.09%)
2.  Brazil	257 (12.36%)	231 (13.52%)
3.  India	188 (9.04%)	158 (9.25%)
4.  United Kingdom	107 (5.15%)	75 (4.39%)
5.  Germany	93 (4.47%)	79 (4.63%)
6.  France	67 (3.22%)	52 (3.04%)
7.  Czechia	54 (2.60%)	32 (1.87%)
8.  Japan	49 (2.36%)	33 (1.93%)
9.  Canada	48 (2.31%)	35 (2.05%)
10.  China	46 (2.21%)	35 (2.05%)

Still Worldwide Interest

Zowe Downloads

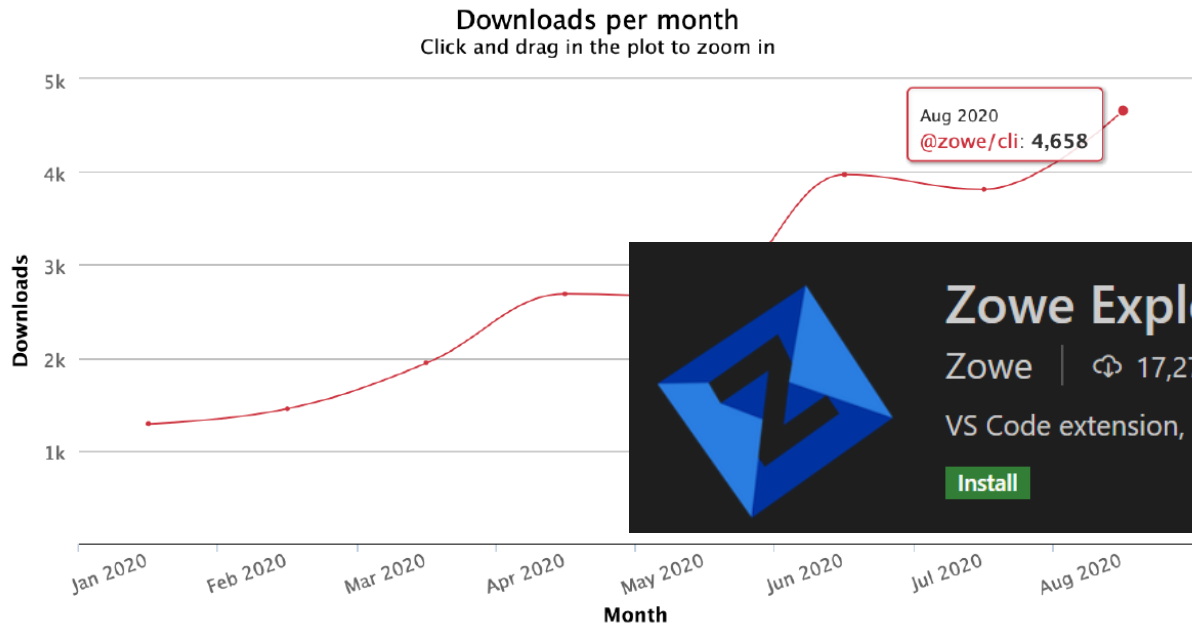
Zowe Explorer on VS marketplace – **13,909** Installs (only cumulative data available)

Zowe.org Cumulative (Beta + v1.X): 10,866 (+550 in June!)



- Ave 510 downloads/month in 2020 of Zowe z/OS components
- CLI “Core” 4,658 in Aug
- Zowe Explorer 17,271 as of Sept 29

Zowe CLI Package Downloads from npmjs.org



Zowe Explorer

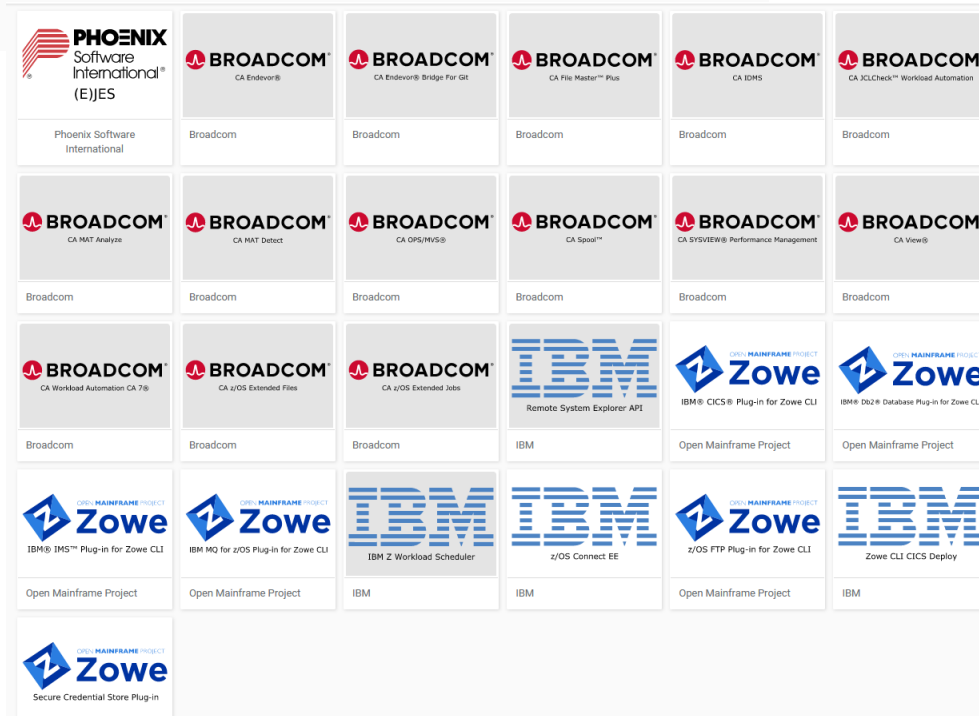
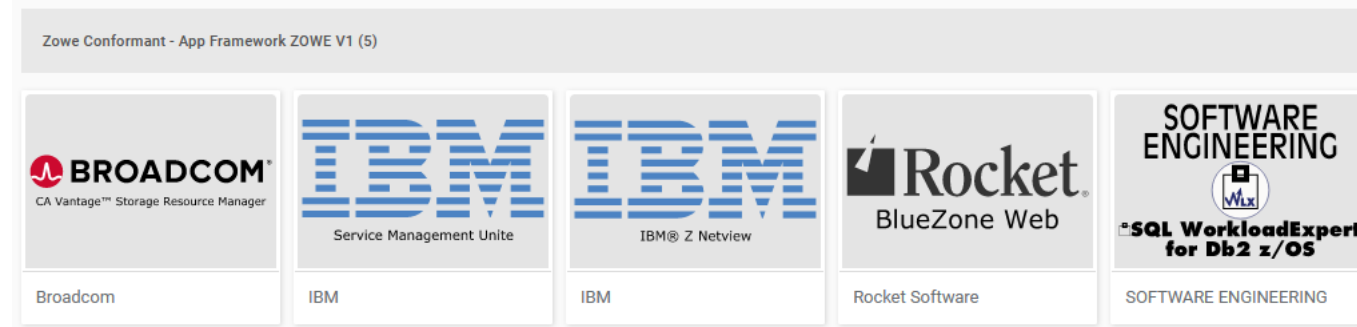
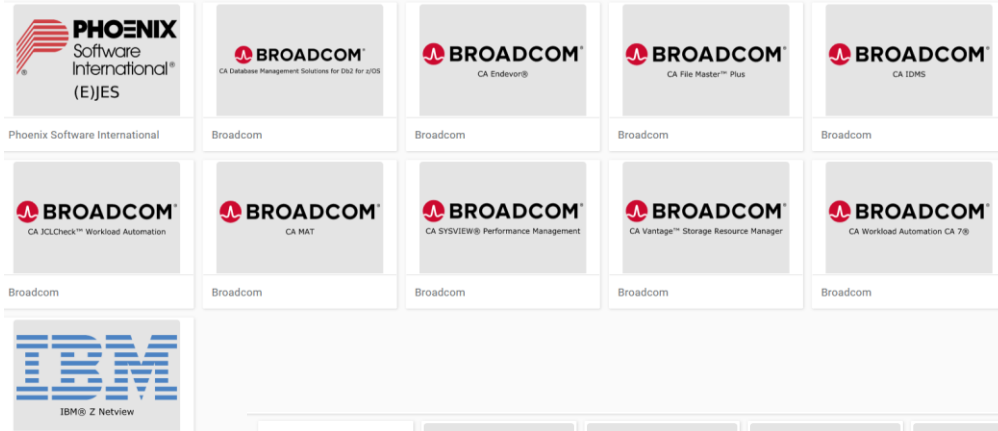
zowe.vscode-extension-for-zowe

Zowe | 17,271 | ★★★★★ | Repository | License | v1.9.0

VS Code extension, powered by Zowe CLI, that streamlines interaction with mainframe data sets, USS files, and jobs

[Install](#)

Zowe (V1) Conformance - <https://www.openmainframeproject.org/projects/zowe/conformance>



Continued increase in conformant extensions

API ML 7 -> 11

App Framework 3 -> 5

CLI 21-> 25

As of Sept 29

SHARE Participation

<https://www.share.org/page/the-latest-news-to-exploit-zowe-in-products-or-your-enterprise-parts-1-3>

3-Part Webinar: The Latest News to Exploit Zowe in Products or Your Enterprise



152 Attendees

A Journey to Zowe - Zowe Conformance and an Extender's journey from learning Zowe to building a Zowe App

Part 1
Begins with an overview of the Zowe V1 Conformance program and will describe how your Zowe component plug-in / extension can earn the V1 Zowe Conformance badge. We'll then walk through the steps that lead a software vendor to choose Zowe and leverage Zowe components as infrastructure for their UI transformation away from Eclipse.

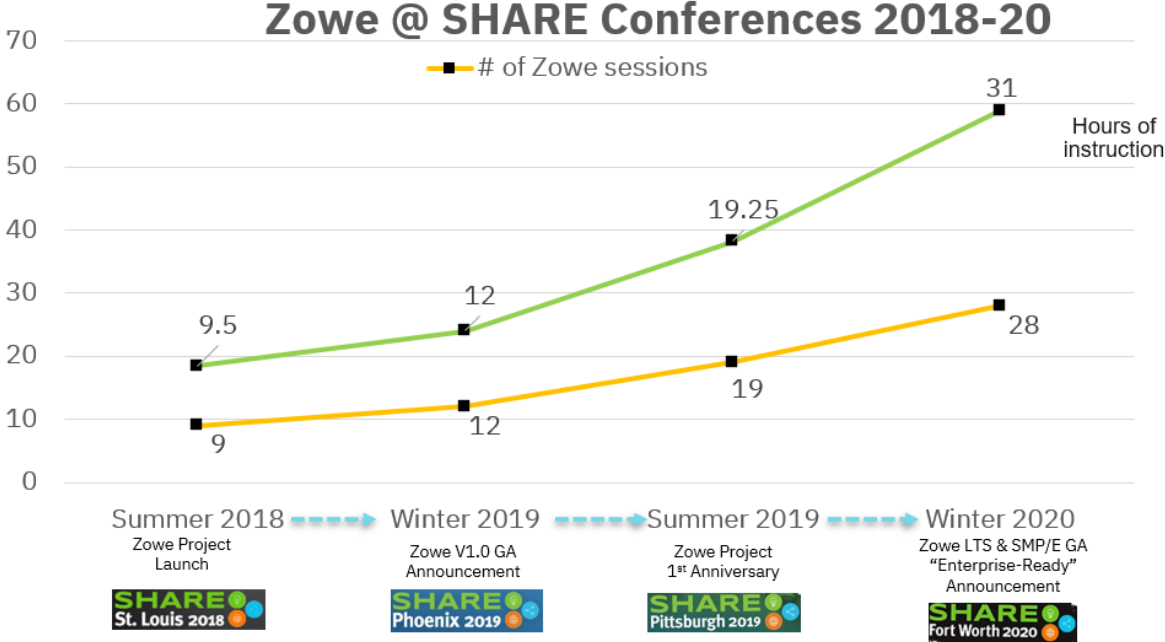
Speakers: Roy Boxwell, Software Engineering GmbH; Bruce Armstrong, IBM; Joe Winchester, IBM - Hursley Labs; Rose Sakach, Broadcom

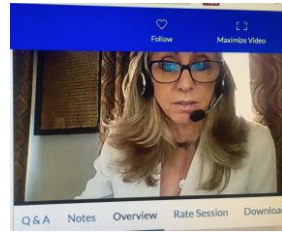
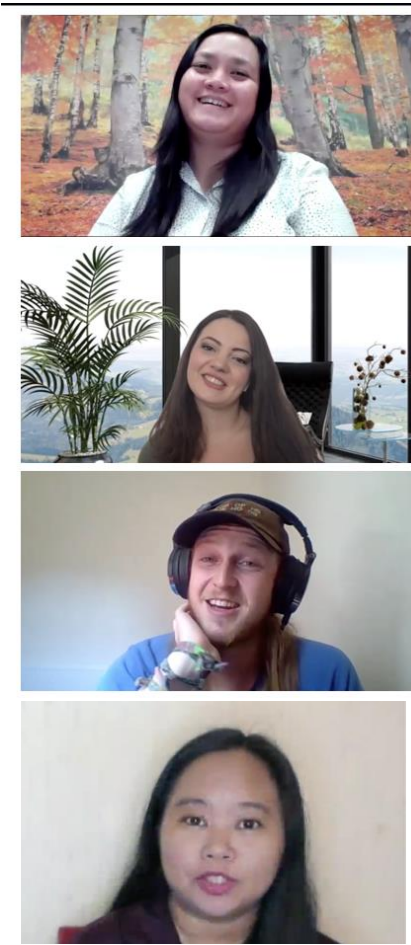
Part 2
Zowe Extensions - Zowe is open source software on z/OS that anyone can use. Part 2 will provide the latest news on Zowe features and functions and explain various ways you can exploit or extend the technology. Zowe provides several out of the box uses that are being enhanced monthly. There are also many extension points to allow you to tailor a solution that meets your needs. Come learn how Zowe can help you organize REST APIs on z/OS, automate tasks with the Command Line Interface or integrate browser applications. You will from the experts writing Zowe and be able to get involved in this revolutionary software for the z/OS platform.

Speakers: Bruce Armstrong, IBM; Joe Winchester, IBM - Hursley Labs; Dan Kelosky, Braodcom; Sean Grady, Rocket Software

Part 3
Zowe 2020 - Part 3 will present a summary of what the open community has accomplished in the first half of the year and describe the development goals for the remainder of 2020. The open community conducted a public planning session in June to describe the project. Come learn the plans and learn how to get involved. We will review ways to participate in the community.

Speakers: Michael DuBois, Broadcom; Peter Fandel, Rocket Software; Bruce Armstrong, IBM





Open Mainframe Project
541 subscribers

Uploads PLAY ALL SORT BY

Cloud Foundry Orchestrated by Kubernetes on Linux on IBM Z	Zowe Conformance: High-reliability Extensions for Mainframe Tools, Guaranteed	OMP Mentorship Project: Summer Internships and Beyond	A 360 Degree View on LinuxONE Security & Compliance	Panel: White Collar, Blue Collar, What is New Collar?
Lightning Talk: How Zowe and Open Source Made Me Talk to the Mainframe (literally)	An Open Source Delivery and Deployment Model for the Security Conscious	Beyond the Mainframe Security Features, It is Time to Learn about Open Source Software Security	Open Source as Common Ground Between the z/OS Enterprise and Everyone Else	Open Mainframe Summit 2020 Speaker Walkthrough
Zowe - What It Means for Mainframe Modernization, the Landscape of z/OS, How It Works, What It Delivers, and Its Future	COBOL and the Modern Mainframe Movement	Leveraging Open Source and Hybrid Cloud for z/OS - A Ruben Goldberg Evolution to COTS Value	Keynote: Building Upon Zowe: How Rocket is Building Commercial Software for Zowe Users	Open Source Tool to Modernize the Build of the z/OS Development, using Jenkins and Git

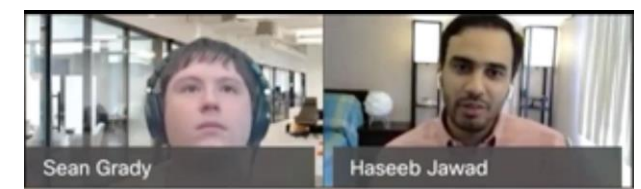
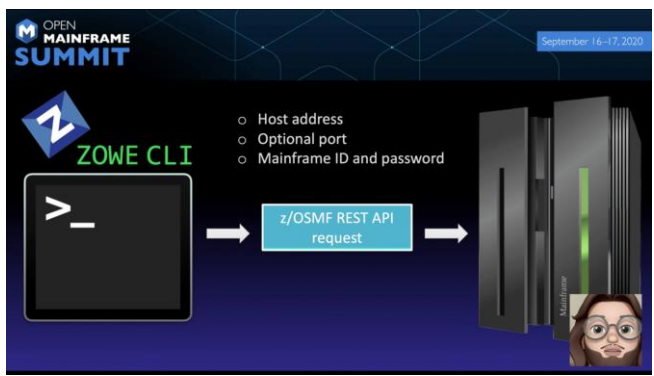
Open Mainframe Project Announces Record Growth with the Launch of Four New Projects, a COBOL Working Group and Micro Focus as a New Member
CBT Tape, GenevaERS, Software Discovery Tool and Mainframe Open Education solidifies Open Mainframe Project as a cornerstone for training, enterprise, devops and z/OS

350+ Registered Attendees

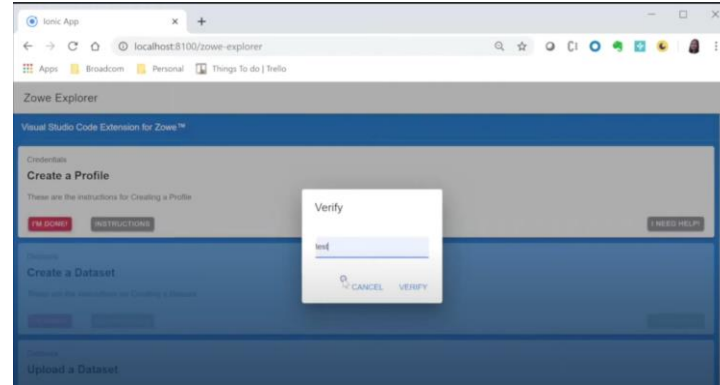
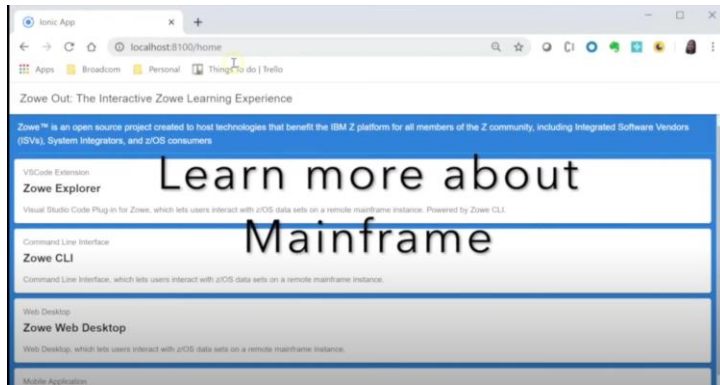
Please rate how the event improved your understanding of the Zowe Project. Very useful (41%) Extremely useful (31%)

How likely are you to start a Zowe project or PoC in the next 6-12 months? Very likely (28%) Extremely likely (31%)

All 37 talks are available on YouTube
19 talks had Zowe content

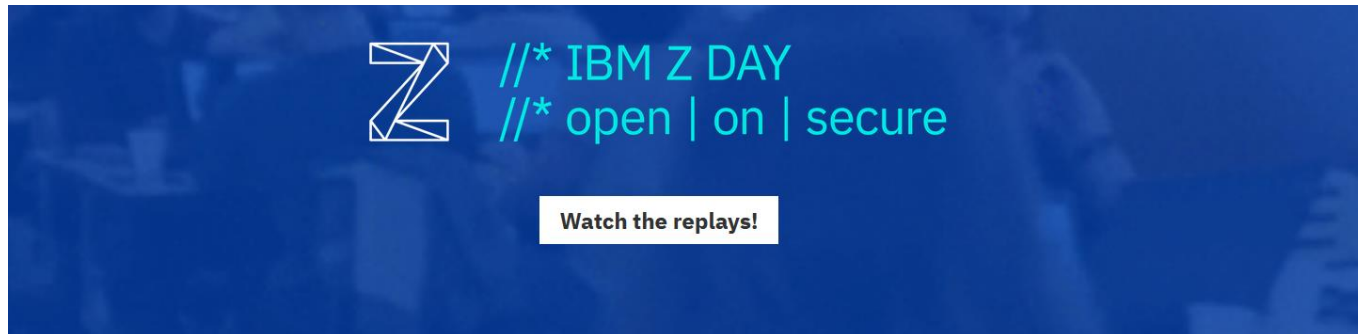


Zowe Hackathon Winners – Zowe Out



https://www.youtube.com/watch?v=IHN_6QNPSP8&t






This year's IBM Z Day was truly bigger and better than ever. On September 15, IBM Z Day launched and shattered IBM virtual community event records, delivering 70 content-rich sessions with more than 100 presenters across **five learning tracks**:

1. [IBM Z](#) – focus on application/data modernization, security, and resiliency
2. [Open Z](#) – the fundamentals of open source work being done on Z
3. [Global Z](#) – non-English-language track
4. [Voice of Z](#) – how IBM Z is being used in the real world
5. [Master the Mainframe](#) – a 24-hour code-a-thon

This free virtual conference drew more than 11,500 registrants and 51,000 live views across all sessions, with participants from more than 70 countries.

15-Sept
8:30 AM

Open Mainframe Project Overview – Open Z Track Keynote
[View more info on keynotes](#)



John Mertic
Director of Program
Management
The Linux Foundation

15-Sept
2:30 PM

Demo: Zowe in action



Thomas McQuitty
Senior Principal Solution
Engineer
Broadcom


Master the Mainframe

It was a fast-paced and thrilling 24 hours for the first-ever Master the Mainframe 2020 code-a-thon and global launch!


Ten student IBM Z Ambassadors coded without a net, live in front of an audience, as they went through Level 1 Master the Mainframe challenges with the help of a mainframe professional.

15-Sept
9:00 AM

Mainframe musings, wishes, challenges, and vision for the future from a customer perspective



Rune Christensen
Senior Software Developer
Bankdata



Joe Winchester
Senior Technical Staff
Member
IBM

Master the Mainframe

Developing enterprise computing and coding skills in more than 3,500 schools in over 130 countries

Get started

MTM used to be taught using 3270 emulators with ISPF, SDSF, ...



```
DATA SETS
├── Z99999.DFSORT.MERGE
├── Z99999.DFSORT.MERGE.COMD
├── Z99999.INPUT
├── Z99999.JCL
│   ├── ADDAMT
│   ├── CBL001
│   └── Z99999.JCL(CBL0001).JCL
└── UNLK.SYSTEM

Z99999.JCL(CBL0001).JCL
1 //CBL0001 JOB 1,NOTIFY=SSYSUID
2 //
3 //COBOL EXEC IGYWCL
4 //COBOL.SYSJOB DD DSN=SSYSUID.CBL(CBL0001),DISP=SHR
5 //LKED.SYSJOB DD DSN=SSYSUID.LOAD(CBL0001),DISP=SHR
6 //
7 // IF RC = 0 THEN
8 //
9 //RUN EXEC PGM=CBL0001
10 //
11 //STEPLIB DD DSN=SSYSUID.LOAD,DISP=SHR
12 //ACCTREC DD DSN=SSYSUID.BATA,DISP=SHR
13 //PKTLINE DD SYSOUT=,OUTLEN=15000
14 //SYSOUT DD SYSOUT=,OUTLEN=15000
15 //CEEDUMP DD DUMMY
16 //SYSDUMP DD DUMMY
17 // ELSE
18 // ENQIF
19
```

OUTPUT

```
Zowe MVS Command
OK 0309 PAGE 58W1, LOCAL
OK 030A PAGE 58W1, LOCAL
OK 0304 PAGE 58W1, LOCAL
0305 PAGE 58W1, LOCAL
030C PAGE 58W1, LOCAL
```

```
DATA SETS
├── UNLK.SYSTEM SERVICES
│   ├── demo
│   ├── demo_bddr
│   ├── demo_hello
│   ├── demo_hello2
│   ├── demo_hello3
│   ├── demo_hello4
│   ├── demo_hello5
│   ├── demo_hello6
│   ├── demo_hello7
│   ├── demo_hello8
│   ├── demo_hello9
│   ├── demo_hello10
│   ├── demo_hello11
│   ├── demo_hello12
│   ├── demo_hello13
│   ├── demo_hello14
│   ├── demo_hello15
│   ├── demo_hello16
│   ├── demo_hello17
│   ├── demo_hello18
│   ├── demo_hello19
│   ├── demo_hello20
│   ├── demo_hello21
│   ├── demo_hello22
│   ├── demo_hello23
│   ├── demo_hello24
│   ├── demo_hello25
│   ├── demo_hello26
│   ├── demo_hello27
│   ├── demo_hello28
│   ├── demo_hello29
│   ├── demo_hello30
│   ├── demo_hello31
│   ├── demo_hello32
│   ├── demo_hello33
│   ├── demo_hello34
│   ├── demo_hello35
│   ├── demo_hello36
│   ├── demo_hello37
│   ├── demo_hello38
│   ├── demo_hello39
│   ├── demo_hello40
│   ├── demo_hello41
│   ├── demo_hello42
│   ├── demo_hello43
│   ├── demo_hello44
│   ├── demo_hello45
│   ├── demo_hello46
│   ├── demo_hello47
│   ├── demo_hello48
│   ├── demo_hello49
│   ├── demo_hello50
│   ├── demo_hello51
│   ├── demo_hello52
│   ├── demo_hello53
│   ├── demo_hello54
│   ├── demo_hello55
│   ├── demo_hello56
│   ├── demo_hello57
│   ├── demo_hello58
│   ├── demo_hello59
│   ├── demo_hello60
│   ├── demo_hello61
│   ├── demo_hello62
│   ├── demo_hello63
│   ├── demo_hello64
│   ├── demo_hello65
│   ├── demo_hello66
│   ├── demo_hello67
│   ├── demo_hello68
│   ├── demo_hello69
│   ├── demo_hello70
│   ├── demo_hello71
│   ├── demo_hello72
│   ├── demo_hello73
│   ├── demo_hello74
│   ├── demo_hello75
│   ├── demo_hello76
│   ├── demo_hello77
│   ├── demo_hello78
│   ├── demo_hello79
│   ├── demo_hello80
│   ├── demo_hello81
│   ├── demo_hello82
│   ├── demo_hello83
│   ├── demo_hello84
│   ├── demo_hello85
│   ├── demo_hello86
│   ├── demo_hello87
│   ├── demo_hello88
│   ├── demo_hello89
│   ├── demo_hello90
│   ├── demo_hello91
│   ├── demo_hello92
│   ├── demo_hello93
│   ├── demo_hello94
│   ├── demo_hello95
│   ├── demo_hello96
│   ├── demo_hello97
│   ├── demo_hello98
│   ├── demo_hello99
│   └── demo_hello100
└── UNLK.SYSTEM SERVICES

script.sh
1 #!/bin/bash
2 files=$(ls)
3 echo "These are the files and folders we have!"
4
5 echo
6 echo
7 echo
8 echo
9 echo
10 echo
11 echo
12 echo
13 echo
14 echo
15 echo
16 echo
17 echo
18 echo
19 echo
20 echo
21 echo
22 echo
23 echo
24 echo
25 echo
26 echo
27 echo
28 echo
29 echo
30 echo
31 echo
32 echo
33 echo
34 echo
35 echo
36 echo
37 echo
38 echo
39 echo
40 echo
41 echo
42 echo
43 echo
44 echo
45 echo
46 echo
47 echo
48 echo
49 echo
50 echo
51 echo
52 echo
53 echo
54 echo
55 echo
56 echo
57 echo
58 echo
59 echo
60 echo
61 echo
62 echo
63 echo
64 echo
65 echo
66 echo
67 echo
68 echo
69 echo
70 echo
71 echo
72 echo
73 echo
74 echo
75 echo
76 echo
77 echo
78 echo
79 echo
80 echo
81 echo
82 echo
83 echo
84 echo
85 echo
86 echo
87 echo
88 echo
89 echo
90 echo
91 echo
92 echo
93 echo
94 echo
95 echo
96 echo
97 echo
98 echo
99 echo
100 echo
```

```
DATA SETS
├── UNLK.SYSTEM SERVICES
│   ├── demo
│   ├── demo_bddr
│   ├── demo_hello
│   ├── demo_hello2
│   ├── demo_hello3
│   ├── demo_hello4
│   ├── demo_hello5
│   ├── demo_hello6
│   ├── demo_hello7
│   ├── demo_hello8
│   ├── demo_hello9
│   ├── demo_hello10
│   ├── demo_hello11
│   ├── demo_hello12
│   ├── demo_hello13
│   ├── demo_hello14
│   ├── demo_hello15
│   ├── demo_hello16
│   ├── demo_hello17
│   ├── demo_hello18
│   ├── demo_hello19
│   ├── demo_hello20
│   ├── demo_hello21
│   ├── demo_hello22
│   ├── demo_hello23
│   ├── demo_hello24
│   ├── demo_hello25
│   ├── demo_hello26
│   ├── demo_hello27
│   ├── demo_hello28
│   ├── demo_hello29
│   ├── demo_hello30
│   ├── demo_hello31
│   ├── demo_hello32
│   ├── demo_hello33
│   ├── demo_hello34
│   ├── demo_hello35
│   ├── demo_hello36
│   ├── demo_hello37
│   ├── demo_hello38
│   ├── demo_hello39
│   ├── demo_hello40
│   ├── demo_hello41
│   ├── demo_hello42
│   ├── demo_hello43
│   ├── demo_hello44
│   ├── demo_hello45
│   ├── demo_hello46
│   ├── demo_hello47
│   ├── demo_hello48
│   ├── demo_hello49
│   ├── demo_hello50
│   ├── demo_hello51
│   ├── demo_hello52
│   ├── demo_hello53
│   ├── demo_hello54
│   ├── demo_hello55
│   ├── demo_hello56
│   ├── demo_hello57
│   ├── demo_hello58
│   ├── demo_hello59
│   ├── demo_hello60
│   ├── demo_hello61
│   ├── demo_hello62
│   ├── demo_hello63
│   ├── demo_hello64
│   ├── demo_hello65
│   ├── demo_hello66
│   ├── demo_hello67
│   ├── demo_hello68
│   ├── demo_hello69
│   ├── demo_hello70
│   ├── demo_hello71
│   ├── demo_hello72
│   ├── demo_hello73
│   ├── demo_hello74
│   ├── demo_hello75
│   ├── demo_hello76
│   ├── demo_hello77
│   ├── demo_hello78
│   ├── demo_hello79
│   ├── demo_hello80
│   ├── demo_hello81
│   ├── demo_hello82
│   ├── demo_hello83
│   ├── demo_hello84
│   ├── demo_hello85
│   ├── demo_hello86
│   ├── demo_hello87
│   ├── demo_hello88
│   ├── demo_hello89
│   ├── demo_hello90
│   ├── demo_hello91
│   ├── demo_hello92
│   ├── demo_hello93
│   ├── demo_hello94
│   ├── demo_hello95
│   ├── demo_hello96
│   ├── demo_hello97
│   ├── demo_hello98
│   ├── demo_hello99
│   └── demo_hello100
└── UNLK.SYSTEM SERVICES

script.sh
1 #!/bin/bash
2 files=$(ls)
3 echo "These are the files and folders we have!"
4
5 echo
6 echo
7 echo
8 echo
9 echo
10 echo
11 echo
12 echo
13 echo
14 echo
15 echo
16 echo
17 echo
18 echo
19 echo
20 echo
21 echo
22 echo
23 echo
24 echo
25 echo
26 echo
27 echo
28 echo
29 echo
30 echo
31 echo
32 echo
33 echo
34 echo
35 echo
36 echo
37 echo
38 echo
39 echo
40 echo
41 echo
42 echo
43 echo
44 echo
45 echo
46 echo
47 echo
48 echo
49 echo
50 echo
51 echo
52 echo
53 echo
54 echo
55 echo
56 echo
57 echo
58 echo
59 echo
60 echo
61 echo
62 echo
63 echo
64 echo
65 echo
66 echo
67 echo
68 echo
69 echo
70 echo
71 echo
72 echo
73 echo
74 echo
75 echo
76 echo
77 echo
78 echo
79 echo
80 echo
81 echo
82 echo
83 echo
84 echo
85 echo
86 echo
87 echo
88 echo
89 echo
90 echo
91 echo
92 echo
93 echo
94 echo
95 echo
96 echo
97 echo
98 echo
99 echo
100 echo
```

YouTube GB bill pereira

Bill Pereira 312 subscribers

Master the Mainframe - USS Challenges 88 views • 4 days ago

Master the Mainframe 2020 - PDS Challenges 100 views • 6 days ago

Getting started on the new Master the Mainframe!! 183 views • 1 week ago

MTM 2020 uses Zowe Explorer

Introducing* the Zowe TSC (Technical Steering Committee)

- Note, the charter is now in draft form. Comments are welcome
- <https://github.com/zowe/zlc/issues/183>

Zowe TSC – What is it for?

- Foster consensus for themes recommended by a clear majority of Zowe consumers, ZLC, or cross Squad initiatives
- Coordinating technical direction of cross-squad efforts
- Running of an architecture forum to explore new ideas for Zowe
- Negotiate and gain consensus of the priority of cross-squad activities with squads (squads retain in-squad backlog prioritization)
- Foster engineering and release best practices
- Define Zowe's release schedule and release readiness criteria
- Approve Zowe releases in accordance with defined criteria

Zowe TSC – Who will be in it?

- Anyone can join and listen in
- Voting members include:
 - Squad leads of the Zowe sub-projects (excepting incubation if distinct)
 - The Zowe Systems Team will be considered a squad for the TSC
 - Other voting representatives will be allowed by consensus of the TSC members (such as Documentation/Web Site administration)
 - Additional Technical Leaders (such as Architects and/or Distinguished Engineers) that are nominated (and 2nd) by TSC members and voted onto the TSC by unanimous agreement
- 1st year only additional technical leaders are determined by ZLC
 - Year 1: Mark Ackert, Sean Grady and Joe Winchester
 - Subsequently, by TSC simple majority vote

Zowe TSC – Why?

1. We want the ZLC to focus more on marketing, recruiting and communication and less on technical coordination
2. It makes no sense for ZLC to approve releases as many of us are divorced from the sausage making
3. When it comes to coordination and prioritization of cross-squad activities, much work is done informally by ZLC and senior technical staff in an ad-hoc manner
4. We want a formal process with responsibility ultimately (though indirectly) driven by the squads
5. The TSC is the next step along the road to cross-squad initiatives started in PI3

New OMP internship project



Alex Kim -
Yongkook Kim <Ykim@vicomnet.com>

Zowe
Embedded
Browser for
RMF,
APIs

Note: ZEBRA is not an official name/project for OMP nor ZOWE yet

ZEBRA Mission Statement

- Provide re-usable and industry compliant JSON formatted RMF/SMF data records so that many other ISV SW and Open Source SW can use IBM Z system data

Problem Statement

- RMF(or SMF) data is a critical data point for IBM Z systems programming, capacity planning and performance analysis, but almost every ISV tools re-generate their way of parsed data in 'closed source' and prevents re-use or sharing knowledge of many years of experiences
- There are some free-of-charge performance analysis/graphics tools based on RMF, but lacking some desired capabilities that many modern open source data analysis tools or very hard to integrate with other open source softwares

Design Goal: Generate open source JSON formatted RMF data thru APIs to share/re-use them so that help system programmers, capacity/performance analyzers use more open source utilities out there, perhaps create knowledge base to share among z performance SMEs and users



Bob, Capacity Planner
OMP Corp's
Data Center Ops Grp

As a user, I want to understand health and capacity of our systems by graphs, as well as feeding and storing data in various formats



Tom, Operations Manager
OMP Financials
Data Center

As a user, I want to manage my mainframe's capacity and make a correct decisions based on capacity trend reports and analysis generated by commonly used tools



John, IT Manager
OMP Enterprise
Storage Systems Group

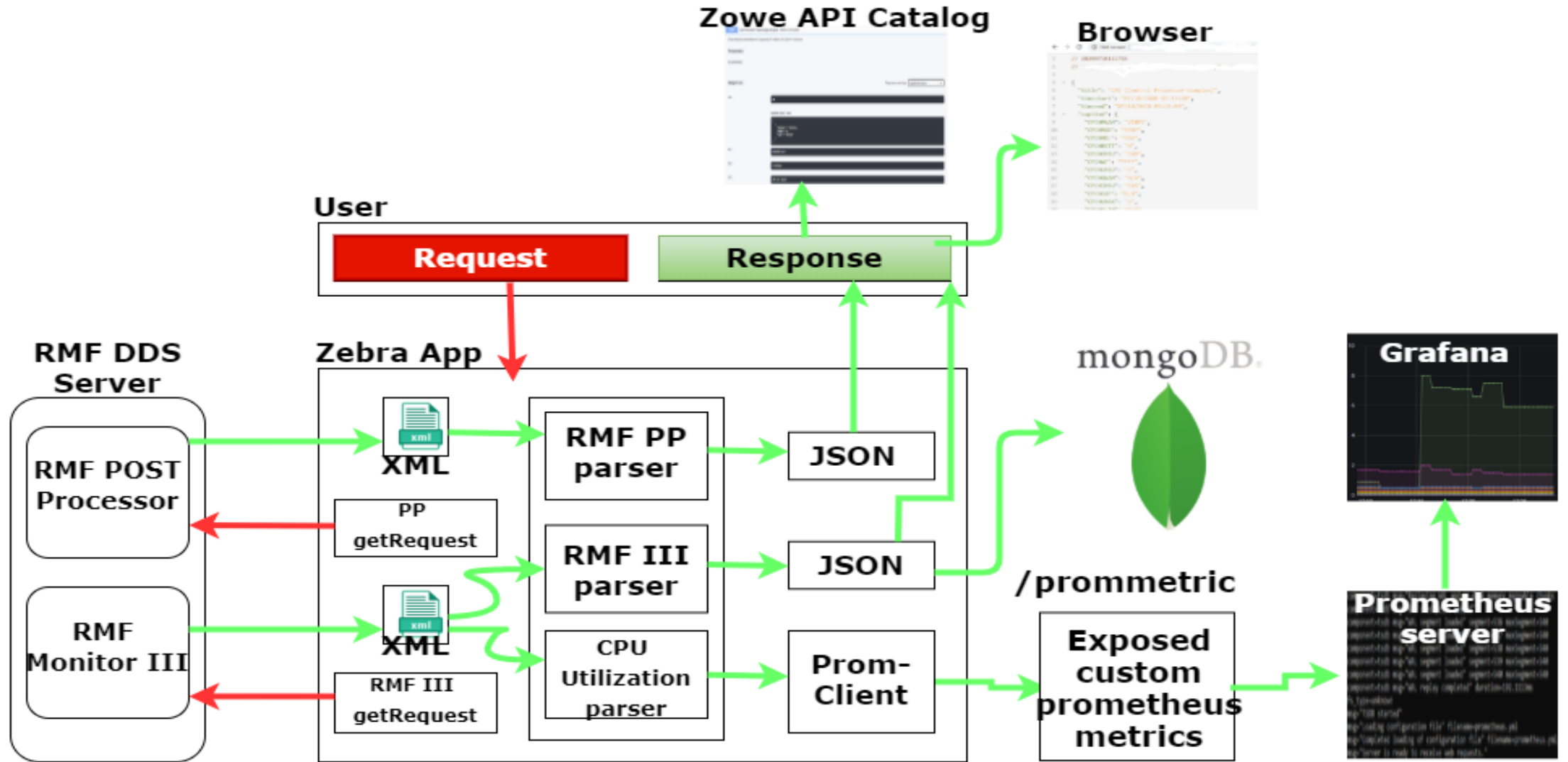
As a user, I want to manage and understand overall storage systems QoS and I/O performance using my own programming and connect to storage management tools



Joann, App Developer
OMP Enterprise Data Center

As a user, I want to create a utility providing single pane of view from multiple architecture server resources using commonly understood data sources

Currently proposed/implemented architecture



ZEBRA with Zowe API Mediation Layer



Search for APIs



Available API services

API Mediation Layer API

The API Mediation Layer for z/OS internal API services. The API Mediation Layer provides a single point of access to mainframe REST APIs and offers enterprise cloud-like feature...

● All services are running

Zebra parsing Engine

RMF Parsing Engine for Monitor I and III Reports

● All services are running

Zebra parsing Engine

RMF Parsing Engine for Monitor I and III Reports

zebra

Zebra parsing Engine

[Service Homepage](#)

RMF Parsing Engine for Monitor I and III Reports

RMF Parsing Engine (ZEBRA)

API Version: 1.0.0

[Base URL: localhost:10010]

Report Parser for RMF Monitor I & III to JSON

[Swagger/OpenAPI JSON Document](#)

RMF Monitor III Convert RMF Monitor III Reports to JSON

GET

/api/rmf3/zebra?report={title}&parm={value} Get RMF III CPC Report by title and caption parameter

API Examples

RMF III Real time monitor

POST {URI}/rmf3/duration?time={time-interval}

GET {URI}/rmf3/duration

GET {URI}/rmf3/CPU/{CPCname}/{LPARname}/{ResourceType}

GET {URI}/rmf3/Workload/{CPCname}/{LPARname}/{ResourceType}

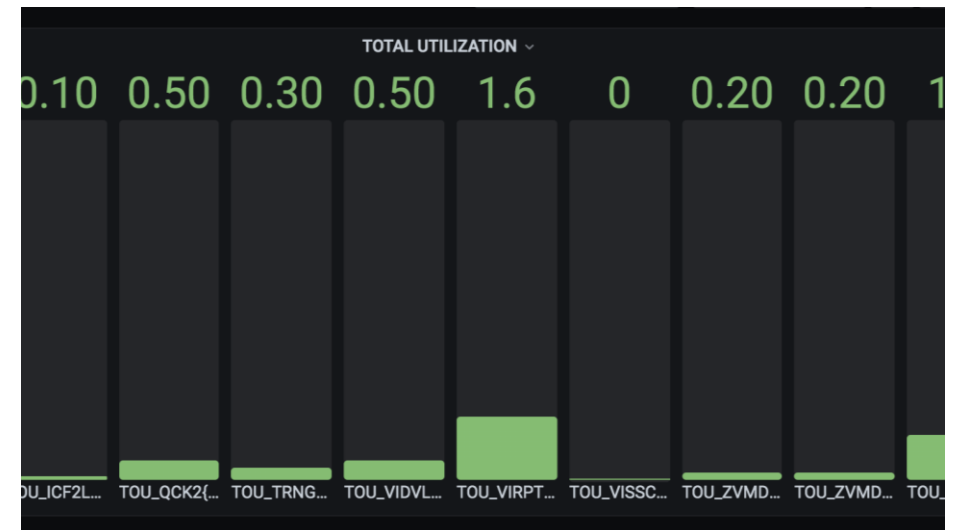
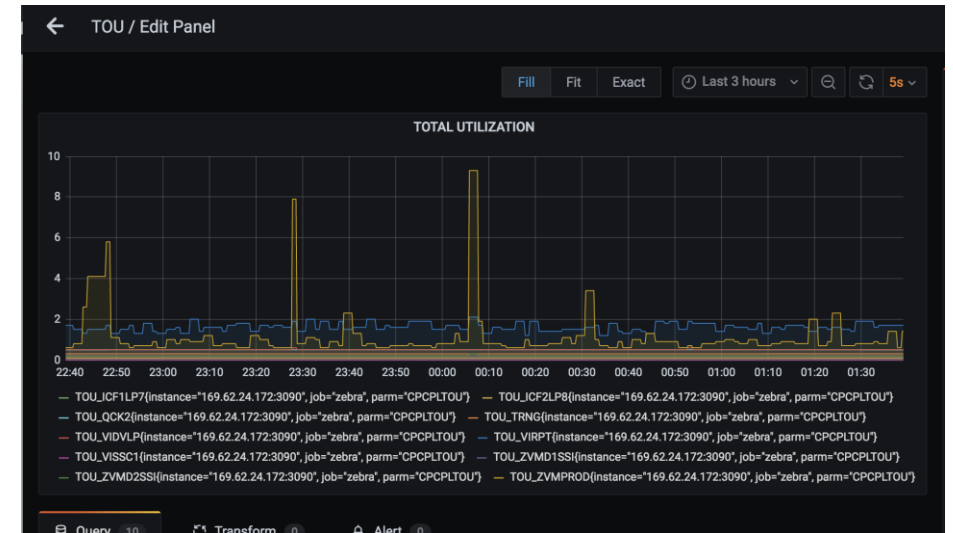
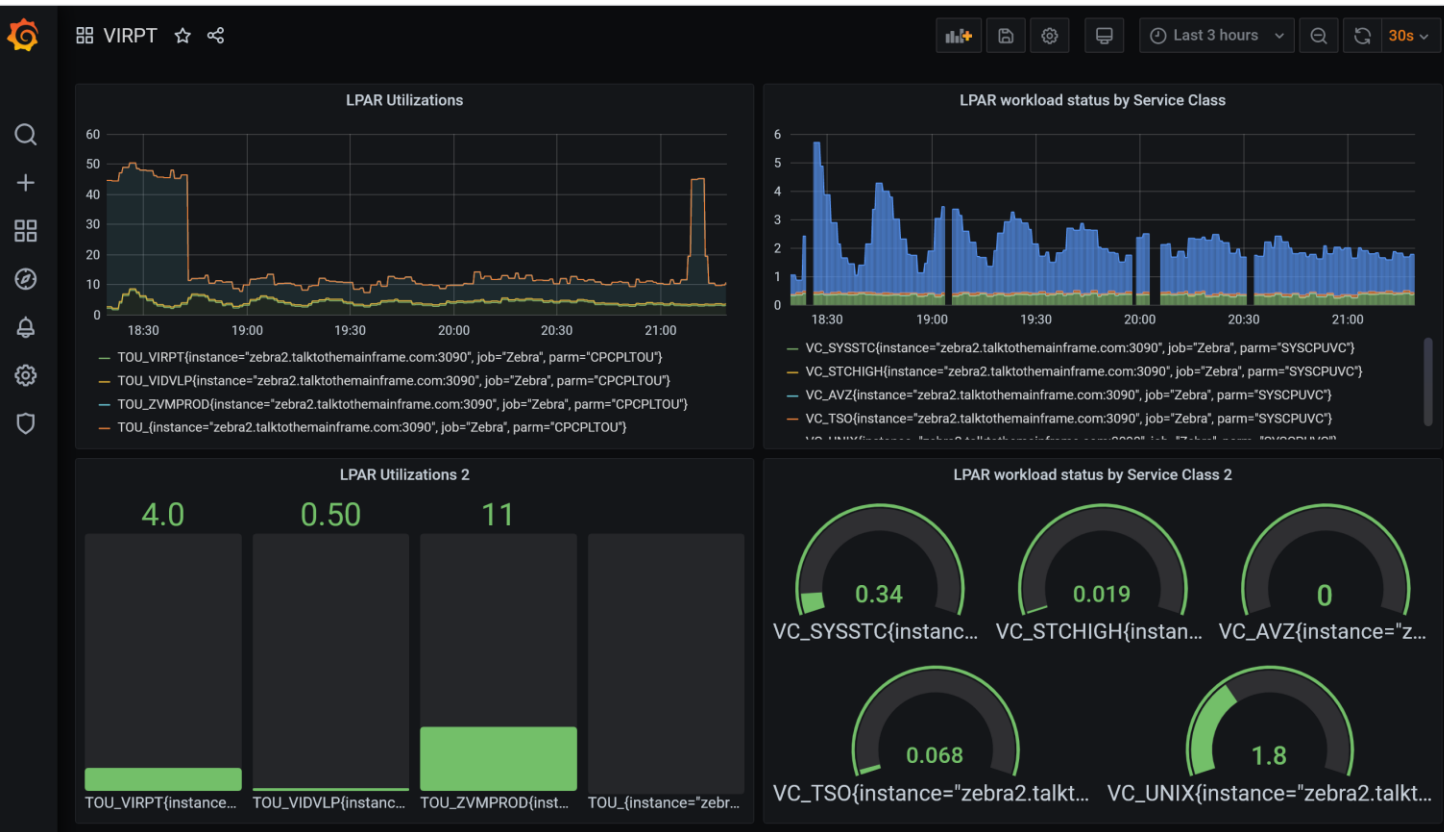
RMF I Static monitor

GET {URI}/rmf/CPU/{CPCname}/{LPARname}/{ResourceType}/{date,start-time, end-time}

GET {URI}/rmf/Workload/{CPCname}/{LPARname}/{ResourceType}/{date,start-time, end-time}

Use Case – feeding data to Grafana

- Either real-time data or historic cpu/workload reports, it can be plotted to various graphics tools. Here one sample showed for CPU LPAR utilization reports



Hills for Zowe Community – Upcoming PI

Systems Team

Performance



- CPU Time Aggregation on SMF30 Records (Continue item from PI3)
- Enhance Performance Test Coverage

High Availability

TSC Assist

- Create Caching API with VSAM support
- Implement Zowe Launcher (Stage 1 defined in HA draft)
- Update Zowe and component packaging to support High Availability
- Verify configuration and PoC on Sysplex
- Add flexibility to define certificate for internal and external usage

CI/CD

- Create plugin installation test based on Ansible playbook
- Automated testing catch-up  *TSC Assist*
- Pipeline improvements (Fix false positives)
- More dashboarding of component, build and test status
- Monitoring of infrastructure - internal (cloud) and external (Marist)
- Stand up more subsystems on Marist - CICS/IMS/MQ/DB2 to allow Zowe CLI extension packages to be tested 
- Improve coverage of tests and resiliency of Zowe pipeline - working with other squads

Zowe Explorer



- Continuation of Conformance Criteria development
- Profile experience improvements
- Improved experience searching and working with data sets and members
- USS-related issues (continuation from 20PI3) - Finish USS renaming
- Selection of UX-related issues (stemming from development in 20PI3)
- Ensure that Zowe Explorer functions properly in a CodeReady Workspace (Subject to access to test environment)

Web UI - Zowe Virtual Desktop and application framework



Containerization

- Launch a beta of both linux & zlinux docker images for zowe server tune, aim to go from beta to full release if feedback positive, + begin work to support zcx

High Availability

TSC Assist

- Finish implementing the auto-restart tool for zowe components
- Implement a state storage tool (vsam based at first)
- Package both into release
- Update servers to utilize state storage tool

Editor unification

- Build MVS/USS explorer features into Editor such that Editor can be a one-stop place for default Zowe Desktop editing experience

App Du Quarter

- App Generator or File Transfer App

Package manager UX improvements

TSC Assist

- Bundle or easily download conda set up public conda server for extended (non-bundled) apps download

ZSS CLI (backlog)

- Make ZSS CLI client

API ML - 20PI4 PI Planning

- x.509 client certificate authentication support for API ML
- API versioning support reflected in the Zowe API ML Catalog
- API ML as a standalone component (Overlap with HA)^{TSC Assist}
- AT-TLS aware API ML
- High-availability support implementation - Caching API and Dynamic Registration related works
- API ML Metrics dashboard




Onboarding - 20PI4 PI Planning

- **Conformance Process Maturity** - Develop a process for updating the Conformance Criteria for all components during ACTIVE LTS, Resolve the incremental Badge debate, and research App-Store
- **Improve initial Onboarding Experience** - Better direct Onboarders to appropriate areas within the Zowe Community
- **Extend OUTREACH efforts** - Increase focus on OUTREACH efforts to Onboard more ISVs and Community members in general
- **Manage Production of and Leverage Statistics** to help all squads to identify Zowe Interest, Experimentation, and Challenges - Continue maturing statistics process and reporting



Zowe Client SDK

- Validate Zowe CLI on Node v14. Node v14 becomes LTS on 10/27.
- Allow for a single profile that stores information commonly needed for core + plug-ins.
- Address growing number of community enhancement requests 
- Ensure documentation ensures successful installation of the Zowe CLI in environments with proxies.
- Ensure Zowe CLI functions properly in a CodeReady Workspace.
- Allow for recently run commands to be easily recalled.

Documentation - 20PI4 Planning

- **Persona/Interest-based doc** - Continued from PI3 - Zowe users can get and browse a list of interested topics within a few clicks by selecting an area of interest, role, and skill level
- **Improve contribution doc** - Continued from PI3 - Zowe contributors can begin contributing to documentation, code, and test by reading the contribution guide on the doc site
- **Various doc enhancements** - Address growing number of doc enhancement requests in GitHub
- **Content analytics** - Analyze the docs web data and user feedback to gather more insights about doc enhancement.
- **Github Wiki Doc Integration** - find an efficient way to migrate information being written on the wiki (by devs) into the doc site.
- **Document the Swift SDK** - Developers can learn about leveraging the Swift SDK (in addition to the Python and Node SDKs)
- **Decide on common format for syntax examples, then implement**
- **Various CLI and API ML specific enhancements**



Reimaging zowe.org home page

- **Add Zowe Explorer**
- **Update terminology to latest description of Zowe**
- **New overview video**
-

Thank You