



Neo generator doser to well know tools like createreact-app (intergration projects from CIUA to generator-neo?)

> [Fab] Move to newer, faster tooling such as vite / esbuild

Build generator neo on top of another tool, like NextJS. Why to build it from scratch when there are other great tools available

> A/B and Canary UI changes [yacine]

Get rid of shims and adjust builds to modern browsers

PT -Typescript
- minimal TSLINT
enforcement
- program to
interface

Improving

readability

of JS errors

in prod

Focus on

documentation

rather than

adding new

features

Webserver architecture

PT - Resource Schema Splitting Shell Into PT - IOC/DI rationalisation to Registry smaller independent enable full validation services that would - something like react Service simplify scalability and stability - programe to interface PT - Decoupling **INeoApplication** Code in Shell Rendering with - avoid is messy services TTL vs. [Atanas] Find a way webRTC annotations to reuse some of the logic which currently lives in the shell. For example NEW Shell entitlements check Cloud Architecture

Roadmap

- Modernize the templates for UI projects
 - · Collaboration with WMPC
 - · Use modern tools to reduce build time

Benefits:

- · Better DX for contributors (less time spent waiting for code to build)
- · Reduce attrition by using latest technologies
- · Bring more self-service to contributors

Roadmap:

- · Review shell architecture once available in Cloud
 - Split the shell into smaller microservices with less responsibilities
- Data analysis of pros / cons of bundling assets vs using http2

Benefits:

- · Increased stability Shell is less a single point of failure
- · Faster release cycle for small microservices
- Decrease inconsistencies between environments (and prod issues related to UI artifacts deployments)
- · Performance improvements in loading Neo UI



Micro frontend

ES6 module support [yacine] Micro FE should be more independent. Reusing the code is good, but it shouldn't crash the apps. Review Single SPA solution on code reusing between Micro FE. There is also a concept similar to artifacts + Module Federation

Roadmap:

· Add es modules support

Benefits:

- Allow multiple panels (for example EVs, workspace) on the same page to load different versions for the same library
- Improve developer experience (less time spent waiting for code to build)



Infrastructure



Have an easy way to do rollout

CDN assets Use Cloud HTTP2 CDN for assets (Images and Is

[Fab] Use of a

CDN to improve

performance of

serving ju/cos

Static

assets

served via

How about HTTP32:) BTW, there is a new HTTP code 103 for static assets



Server Side Rendering

[Fab] Allowing partner teams to have their pages rendered on server side

Microservice for SSR

Prefetch

data on the

server [yaci

ne]

Find a way to be able to use NextIS for SSR

> Somehow integrate NextJS into our UI

Allow using tools such as NextJS to build SSR React applications More SSR in Neo. Potentially it would require separate services deployments

Jamstack



Lazy loading and code splitting (es modules)

Create some way to request and cache api responses to avoid 50 requests for the same thing in a page



React 18 new capabilities (including server side streaming)

[Fab] React 18 upgrade + future upgrades

[Atamas] Make it super easy for apps to move to new versions of

[Atanas] Upgrade the EV to the latest version of the webskd components and react

> Unbundle /remove neo-core-is [Yacine]

Roadmap:

- · Ability to spawn short lived Neo that reflects a given environment, to be used during CI/CD
- · Using HTTP2/3 once we are in Azure

needed)

· Gather data on benefits of using CDN with the goal of hosting our assets on a CDN

Benefits:

- short lived env:
 - · Better integration against prod like env, resulting in shift left testing approach
 - Faster feedback on developer changes
- HTTP2/3 + CDN:
 - · Decrease load on shell and proxy
 - · Decrease load time of Neo UI
 - Improve client experience





Framework agnostic UI components

[Alamas] Do not create any more instances of components :)

[Fab] Should we move to bundling everything (components?) Impact on EVs?



Low-code / no-code

PT - Global Neo FDC3 - lib standard compliant - open to extension global taxonomy of extendable types uses web rtc for comms under the hood? - bridge to desktop

[Fab] Provide a low-code environment

Make ACQA webapps a core Neo platform offering



[Fab] PWA explore again (2 years later)

PT - OpenFin Platform API for Neo Desktop/externalise OneDesk

Other

Recognition and connecting apps with SWC (AppDir) codes

Clean up entitlements we will need to do this for A/B testing?:(

[Fab] mobile version - is there anything we can do to provide a better experience?