Architecture Navigator Knowledge Transfer

This page has been set up and created by NICHOLAS KARIMI MUIGAI, ArchNav mentee (Profile) to document his experiences and understanding of the project as he interacts with it. Therefore, consider this not as documentation for the ArchNav tool but a knowledge-sharing repository.

Problem Statement

In an environment where we had an ideal Wiki and an ideal ReadTheDocs, a small team would be effective in maintaining an up to date documentation, while developers and end-users who want to look up information on documentation would do it effortlessly.

Currently, the Wiki is not structured. There are two sources for ONAP information, ***the developer wiki,*** and the ***ReadTheDocs***. This is a problem because some of the documentation on the wiki is not quickly moved to the ReadThe Docs where essentially it is supposed to reside and gets outdated pretty quick.

This makes it very difficult to find information and find the most current and relevant one. With so much information scattered in the wiki, minimizes value to the users and they will tend to forgo looking up for documentation they were after.

The Architecture Navigator seeks to solve this problem by creating a visual presentation layer with clickable objects representing each individual project under ONAP. The tool aims to offer an interactive interface where users would need to hover over different objects mapping links to the project documentation.

Use Case

The specific use case being addressed by this evaluation is that of providing a method for image-based navigation to the correct formal doc set for developers, implementers, operators, and end-users of ONAP.

What is Architecture Navigator?

Architecture Navigator is a web-based tool that provides common access to the wiki and the Read The Docs documentation. It uses existing project diagrams and overlays the diagrams with clickable are maps. A clickable area object within the diagram is a visual representation of the individual ONAP projects which have hyperlinks embedded on them such that, when a user hovers a mouse over the object and clicks, they are redirected to the project documentation or the wiki.

The Architecture Navigator does not create documentation but allows users to easily navigate through documentation that resides in the Read The Docs.

ArchNav as a Framework

ArchNav is an architectural platform that provides common access to the wiki and RTD. It uses existing ONAP projects diagrams and overlays them with clickable area maps. Its usefulness has been demonstrated beyond ONAP documentation needs thus the tool can be repurposed and used for any other project in LFN.

ArchNav has no static HTML pages per se, all the HTML pages are dynamically created in real-time based on the user request. It uses a file system DB to store JSON objects which on a single modification, can entirely point to a different project.

Current Version

Project	Architecture Navigator
Release Name	dn
Release Version	3.0

Technical Details

- ArchNav does not have any static HTML pages. All HTML pages are dynamically created in real-time based on the "Path" component in the URL request.

    The path drives menu creation.

- ArchNav uses a filesystem DB to store all JSON objects.
- Area maps are defined using a common JSON object.
- Developed using PHP, JSON, CSS, JS
- Apache Webserver

Design Overview

As mentioned in the technical details, ArchNav HTML pages are dynamically created. A JSON object is defined in the notion of topics and sub-topics.

Topic

A topic is defined as a major project such as 5g-blueprint, o-ran e.tc

Sub-topic

Subtopics are components that are necessary to the Topic.

Other Information:

- link to current source code: https://github.com/ca2853/onapdocs/tree/dev

How does ArchNav Work?

- [x] NICHOLAS KARIMI MUIGAI complete evaluation by 30 Jul 2021

Side by side comparison between ArchNav and Read The Docs

This side by side comparison targets the following areas:

1. Functionality
2. Maintenance
3. Infrastructure requirement

	ArchNav	**Read The Docs**
FEATURES	Supported(**Y/N**)	Supported(**Y/N**)
Web-based	**Y**	**Y**
Software Model	standalone server	**SaaS**
page creation	dynamic	static
browser compatibility	Works best on Mozilla Firefox	-browser-independent

FUNCTIONALITIES

Architecture Navigator	**Read The Docs**
Technologies used include: • php • html • javascript • json • css	Technologies used; • Python • Sphinx • reStructuredText
Works on the browser	Works on the browser
Does not create documentation	Creates documentation
Offers interactive overlay	Links embedded in the image.
Content is dynamically loaded as the user navigates to new pages.	Content not dynamically created
Offers quick access to documentation in a click.	Documentation is scattered and does not offer a straightforward way of looking up information.
References links to both formal and development wiki	Holds complete and formal technical documentation.
Clickable maps in the form of image overlay.	Clickable maps in the form of an SVG diagram.

Dynamic code generation	N/A
Allow toggling on and off for some features on demand	N/A
Image creation from any source (bitmap image)	Requires image editing/modification

INFRASTRUCTURE REQUIREMENTS

	ArchNav	Read The Docs
Operating System	Linux	**SaaS**
Web-server	Apache	
vCPU Cores	2	
Memory	8 Gb	
Storage	1 Gb	

MAINTENANCE

ArchNav	Read The Docs
Requires brief end-user training on how to use the tool	Straight forward
Will require a team of technical code contributors/committers	Requires knowledge of rst syntax and Inkscape.
Require manual update of the links to point to the current documentation	**TBD - confirm whether automatic sync with git**
Has multiple moving parts	Fewer moving parts
New features coded by the ONAP community	New features coded by RTD community
Initial setup of new infrastructure and ongoing support	SaaS

FINDINGS

ArchNav is a highly versatile platform that supports a wide variety of use cases. One of the drivers for its creation seems to be in part as a workaround to existing documentation policies that are not being followed, (see DeveloperWiki and ReadTheDocs Usage Policy) The confusion created by not following the set documentation policy in all situations, in addition to the wiki being poorly structured, has created the need for something like ArchNav to help the development community find the information they need. However ArchNav brings complexity when designing a clickable image~~,~~ The process requires taking a snapshot of an existing image, converting it to a PNG file (if not already a PNG), using an external tool to draw shapes around objects in the image to generate the needed coordinates and lastly to store the coordinate attributes in a JSON object. This has the potential to be more time-consuming and error-prone when compared to creating OVERLAYS using an SVG editor to create diagrams with embedded links. SVGs with embedded links however have less versatility than ArchNav has demonstrated.

As a platform ArchNav is not a resource or storage hog since most of its files are generated dynamically on demand using PHP and Javascript and none of the items linked from an image reside locally.

Documentation residing in the read the docs is always up to date and the most current. The ArchNav will require a separate manual update of the links pointing to the documentation in addition to the current RTD updates whenever there is a new release of ONAP.

Moving ArchNav out of the lab to production will have a significant cost implication to the Linux Foundation Networking. The application will require to be run on cloud server instance with resource requirements as oultlined on the (Infrastructure requirements) section above.

RECOMMENDATION

- Implement a FACADE design pattern

As is, the tool can be supported by people with the right technical skillset, however, to broaden the scope and make it accomodating for contribution from technical and non-technical members of the community, a simple web page with input forms as an interface that is mapped to the JSON object instances. In essence, this will eliminate the need for directly modifying the backend code to make changes.

- Implement webhooks on GitHub with the wiki configuration.

To address the concerns for maintaining an up-to-date documentation reference, exploring and experimenting with some automation methods would be ideal. In this case, webhooks can be configured on GitHub to be triggered to make an update to the flat file storage such as when there is a new release or a release tag changes.

Recommendation based on the Use Case

Based on the above findings, the ArchNav platform has to a large extent solved the highlighted shared problem on the problem statement. The goal for any software program is to eliminate complexity and not introduce one. Architecture Navigator on one side which is, offering intuitive graphical navigation, and quick access to the documentation, has consistently produced stellar performance. Conversely, the platform which is a stand-alone application fails in providing an integration structure for providing a single entry point for the ONAP documentation. Additionally, a pain point to be of concern would be the complexity that might arise in implementing clickable image features.

That said, to eliminate any confusion that could arise when ONAP community members try to access the official documentation, ArchNav as a parallel option would not be the ideal solution. However, the functionalities of the ArchNav can be more useful to the opensource community as an alternative to quick navigation and access to formal and development project information. Case in point, the ArchNav platform would be an ideal entry point for the development wiki where developers seeking to navigate the unstructured wiki would be presented with an interactive image-based layer that directs them to the projects development wiki in a single click. To facilitate something like this it would really be best to have ArchNav as an open-source project with the appropriate governance applied to attract the appropriate maintainers and contributors.

Ultimately any documentation model should fit seamlessly into the existing support model and infrastructure for the documentation itself. As such managing, all of the content in the documentation repo in a form that does not require a programming background seems most appropriate to ensure maintainability.

Conclusion

The point here was to highlight the functionality and features that the two tools offer to the end-user. With an objective for providing quick navigation to the documentation based on visual presentation, it would be ideal to have the image-based navigation implemented within the official documentation. Having the application of image navigation within the documentation will ensure that the entry point to the official documentation remains unchanged hence providing a consistent and improved user experience unlike having community members access the documentation through a third-party application. This despite solving the commonly shared problem which is the lack of a simple and intuitive way for accessing the documentation, would bring more confusion of duplicity. For the use case in the original problem statement, ArchNav is probably not the correct solution. However, it may be an entirely appropriate solution to use ArchNav as a visual entry point to areas where information is less structured than our formal documentation set or we do not have full control or management over the source of the documentation being sought.