DocX Resources

Release 0.7.0

Veritas Technologies LLC

Apr 07, 2018

CONTENTS

DocX is the agile shared document development methodology used to develop all documentation related to Multi-Cloud Data Management Platform. You may wish to refer to DocX Developer Resources in addition to the DocX Process Guide and the DocX Style Guide listed below.

CHAPTER

ONE

MCDMP SHARED DEVELOPMENT OF DOCUMENTATION

1.1 What's in a name

The very first thing we'd like to do as part of MCDMP Shared Doc Development Process building, is to rename TechDocs. The reason being, Rohini thinks that the name should not get confused with InfoDev as this effort is not to replace InfoDev but to augment it and strengthen it. The TechDocs initiative is to bring in agile documentation practices from contemporary open source world, leaders in IT into Veritas documentation efforts.

You can participate in TechDocs renaming exercise. Simply click this link TechDocs Rename Survey, and vote. If you don't like any of the alternatives listed there, suggest your own.

1.2 Objective

MCDMP Platform is comprised of a set of micro-services that work in tandem to offer rich functionality for information management. The overall objective is to be the Platform of choice for Enterprise Information Management solutions from Veritas but also third parties. MCDMP allows end users to utilize its rich functionality through REST APIs. To facilitate the onboarding process, MCDMP TechDocs proposes a shared documentation development process similar to the one followed successfully by several popular open source software projects.

This document outlines MCDMP process for shared development of technical documents including the following:

- Getting Started Guide
- Installation Guide
- Concepts and User Guide
- SDK API Documentation

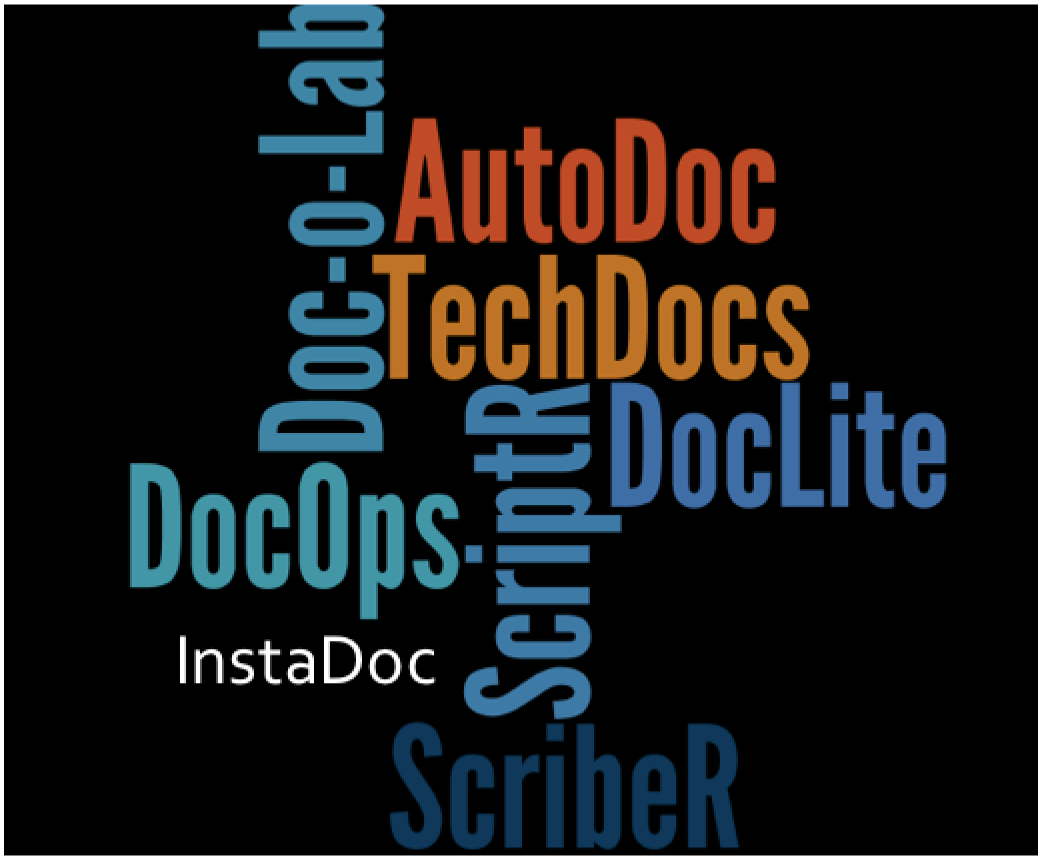
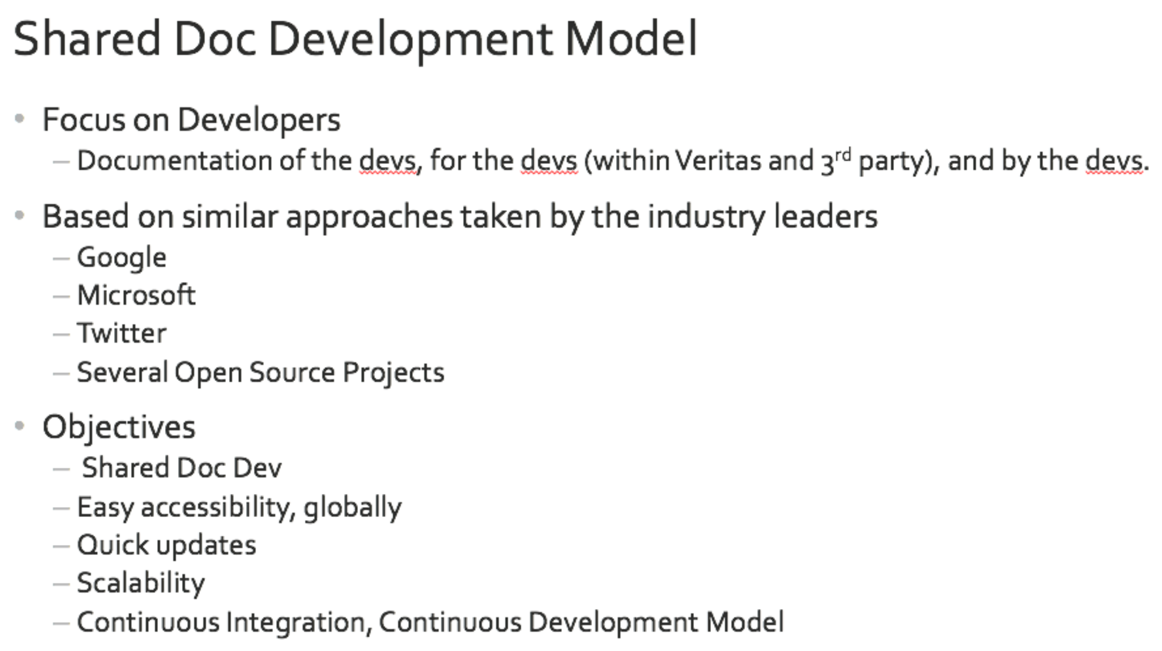


Fig. 1.1: Figure: Some alternative names for TechDocs

- MCDMP FAQ
- Others (TBD)

1.3 Shared Doc Development Process Highlights

Here are some of key highlights of MCDMP Shared Documentation Development Process. Please note, this is a draft as of now and we hope to refine it further in the days to come, as we build MCDMP itself.



Shared Doc Development Model

- Focus on Developers
  - Documentation of the devs, for the devs (within Veritas and 3rd party), and by the devs.
- Based on similar approaches taken by the industry leaders
  - Google
  - Microsoft
  - Twitter
  - Several Open Source Projects
- Objectives
  - Shared Doc Dev
  - Easy accessibility, globally
  - Quick updates
  - Scalability
  - Continuous Integration, Continuous Development Model

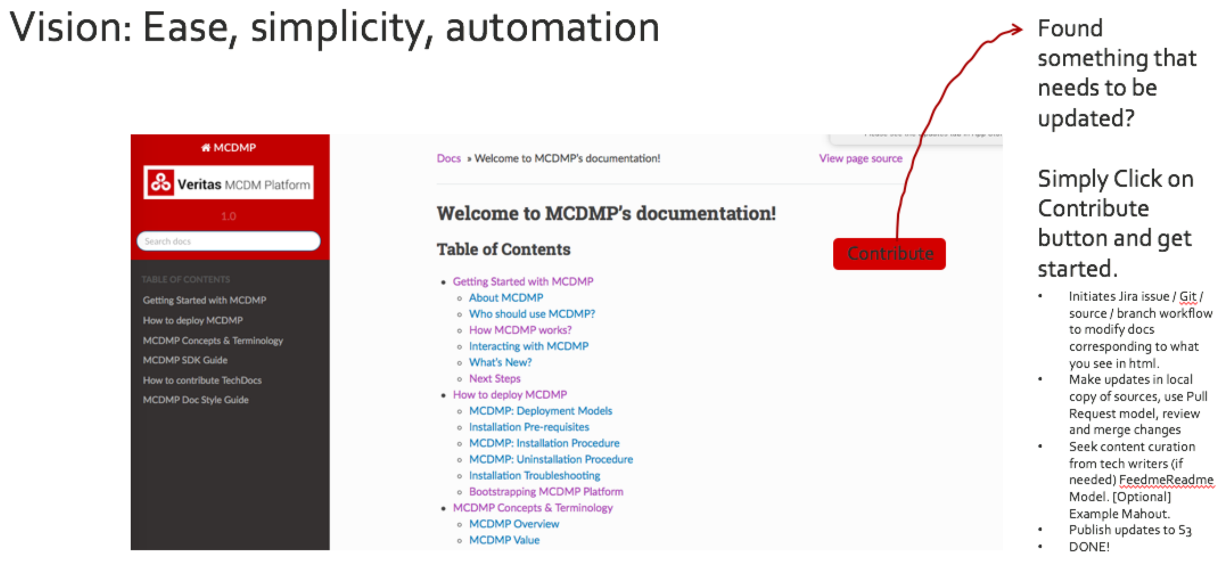
Fig. 1.2: Figure: Shared Document Development Model for MCDMP

1.4 How it works

The following figure is a placeholder for now. It shows how the MCDMP Shared documentation development process is intended to function.

1.5 How to contribute

In this section, we list some high level bullet points on how various MCDMP personnel can contribute to the document content development initiative as we all progress along building MCDMP itself.



Vision: Ease, simplicity, automation

MCDMP

Veritas MCDM Platform

1.0

Search docs

TABLE OF CONTENTS

Getting Started with MCDMP
How to deploy MCDMP
MCDMP Concepts & Terminology
MCDMP SDK Guide
How to contribute TechDocs
MCDMP Doc Style Guide

Docs » Welcome to MCDMP's documentation!

View page source

Contribute

Welcome to MCDMP's documentation!

Table of Contents

- Getting Started with MCDMP
  - About MCDMP
  - Who should use MCDMP?
  - How MCDMP works?
  - Interacting with MCDMP
  - What's New?
  - Next Steps
- How to deploy MCDMP
  - MCDMP: Deployment Models
  - Installation Pre-requisites
  - MCDMP: Installation Procedure
  - MCDMP: Uninstallation Procedure
  - Installation Troubleshooting
  - Bootstrapping MCDMP Platform
- MCDMP Concepts & Terminology
  - MCDMP Overview
  - MCDMP Value

Found something that needs to be updated?

Simply Click on Contribute button and get started.

- Initiates Jira issue / Git / source / branch workflow to modify docs corresponding to what you see in html.
- Make updates in local copy of sources, use Pull Request model, review and merge changes
- Seek content curation from tech writers (if needed) FeedmeReadme Model. [Optional] Example Mahout.
- Publish updates to S3
- DONE!

Fig. 1.3: Figure: Vision: Shared Document Development Model for MCDMP

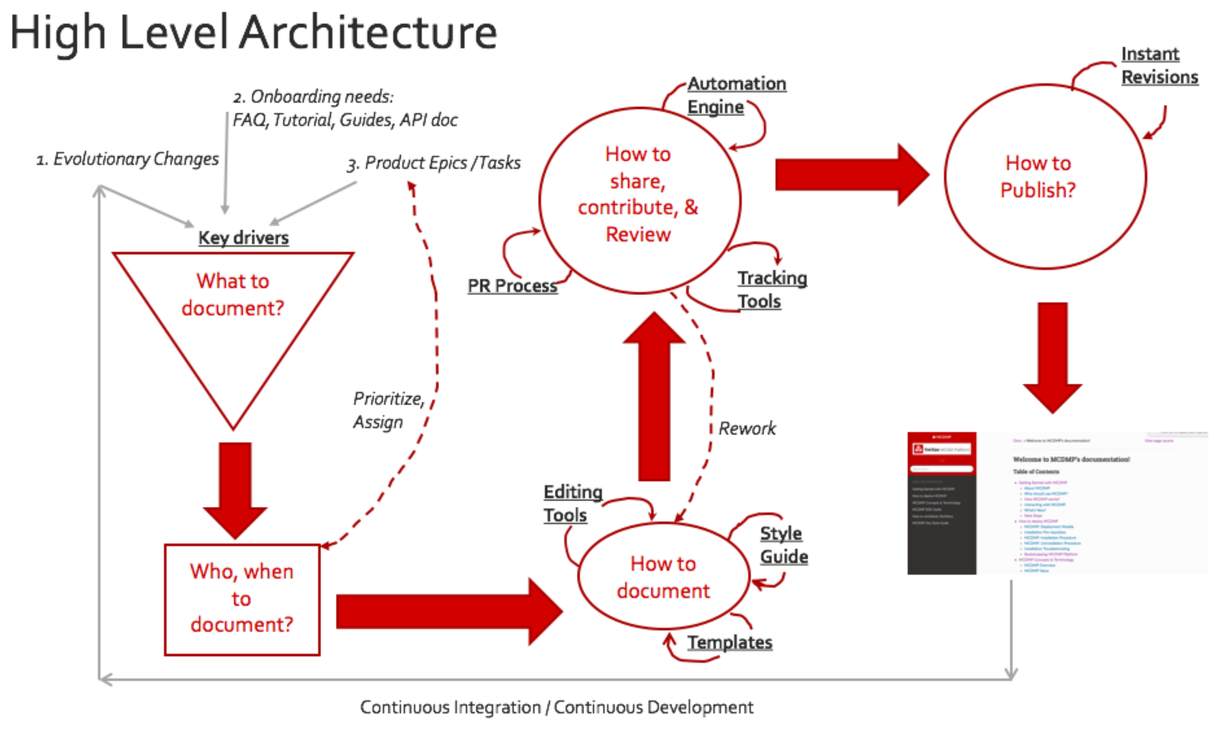
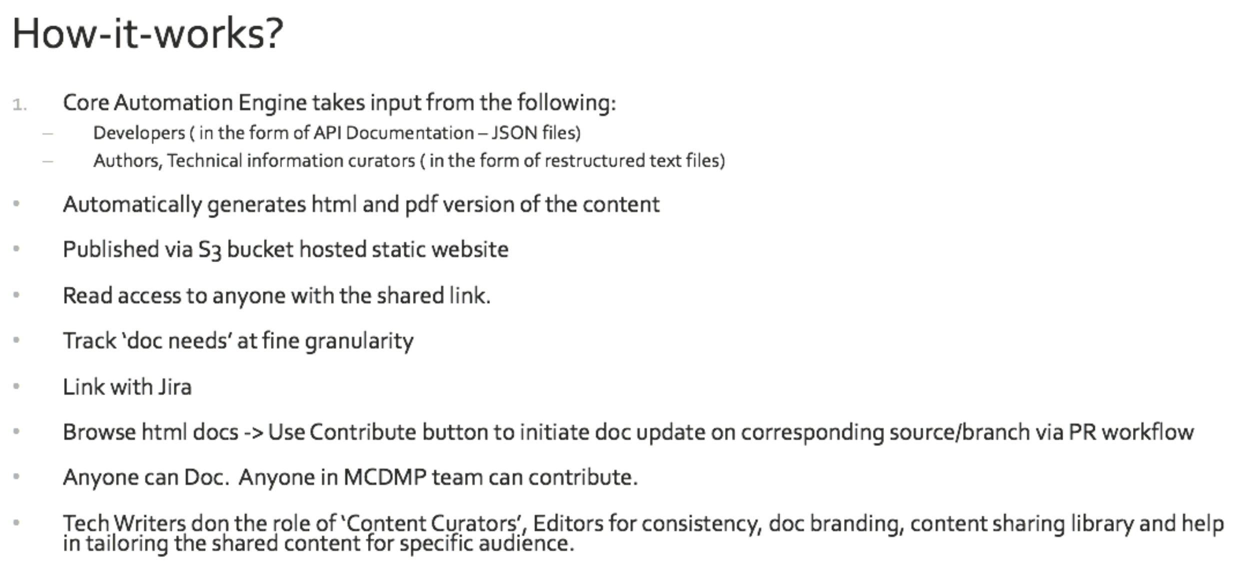


Fig. 1.4: Figure: Architecture: Shared Document Development Model for MCDMP

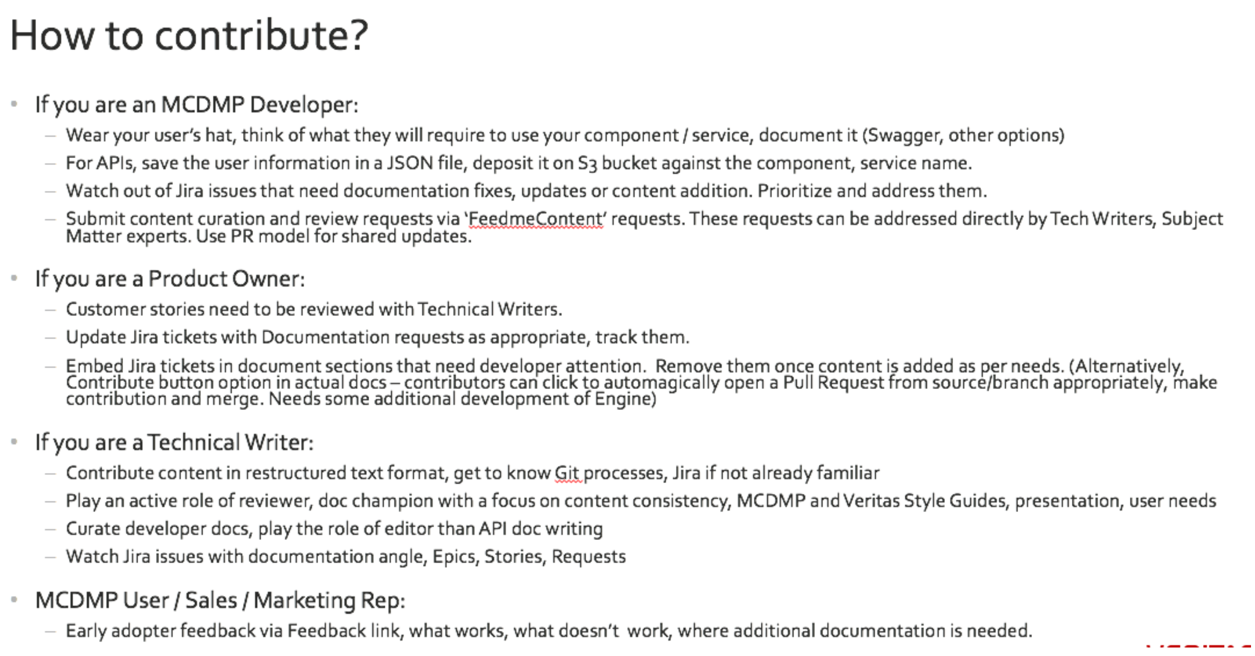


How-it-works?

1. Core Automation Engine takes input from the following:
   - Developers (in the form of API Documentation – JSON files)
   - Authors, Technical information curators (in the form of restructured text files)

- Automatically generates html and pdf version of the content
- Published via S3 bucket hosted static website
- Read access to anyone with the shared link.
- Track 'doc needs' at fine granularity
- Link with Jira
- Browse html docs -> Use Contribute button to initiate doc update on corresponding source/branch via PR workflow
- Anyone can Doc. Anyone in MCDMP team can contribute.
- Tech Writers don the role of 'Content Curators', Editors for consistency, doc branding, content sharing library and help in tailoring the shared content for specific audience.

Fig. 1.5: Figure: How it works: Shared Document Development Model for MCDMP



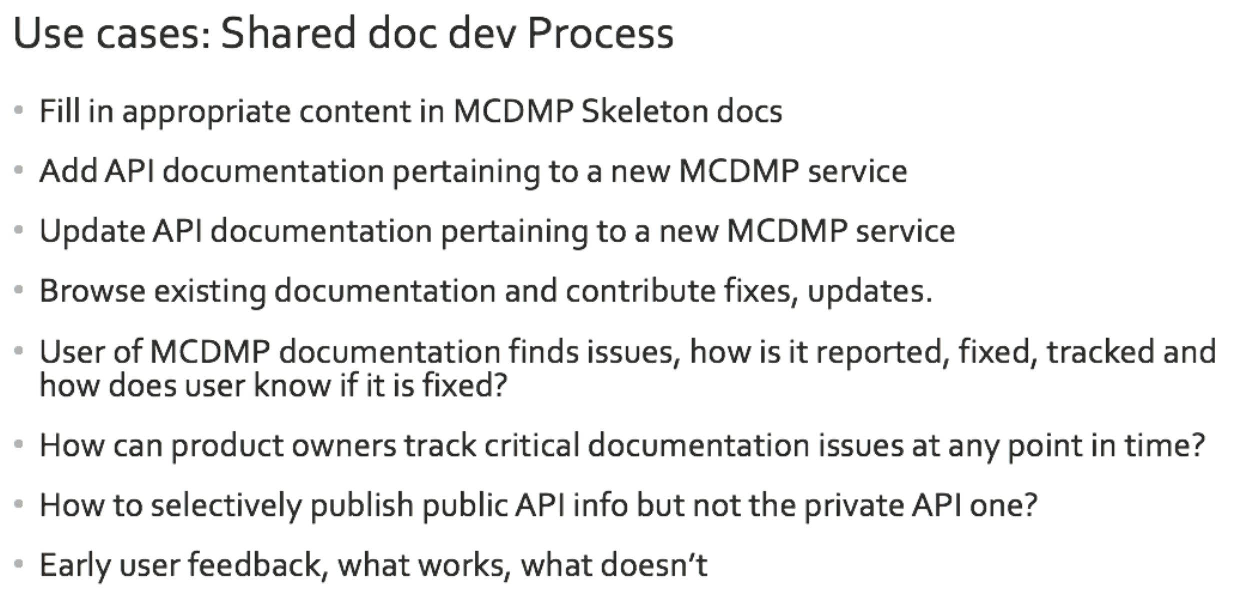
How to contribute?

- If you are an MCDMP Developer:
  - Wear your user's hat, think of what they will require to use your component / service, document it (Swagger, other options)
  - For APIs, save the user information in a JSON file, deposit it on S3 bucket against the component, service name.
  - Watch out of Jira issues that need documentation fixes, updates or content addition. Prioritize and address them.
  - Submit content curation and review requests via 'FeedmeContent' requests. These requests can be addressed directly by Tech Writers, Subject Matter experts. Use PR model for shared updates.
- If you are a Product Owner:
  - Customer stories need to be reviewed with Technical Writers.
  - Update Jira tickets with Documentation requests as appropriate, track them.
  - Embed Jira tickets in document sections that need developer attention. Remove them once content is added as per needs. (Alternatively, Contribute button option in actual docs – contributors can click to automagically open a Pull Request from source/branch appropriately, make contribution and merge. Needs some additional development of Engine)
- If you are a Technical Writer:
  - Contribute content in restructured text format, get to know Git processes, Jira if not already familiar
  - Play an active role of reviewer, doc champion with a focus on content consistency, MCDMP and Veritas Style Guides, presentation, user needs
  - Curate developer docs, play the role of editor than API doc writing
  - Watch Jira issues with documentation angle, Epics, Stories, Requests
- MCDMP User / Sales / Marketing Rep:
  - Early adopter feedback via Feedback link, what works, what doesn't work, where additional documentation is needed.

Fig. 1.6: Figure: How to Contrbute: Shared Document Development Model for MCDMP

1.6 Use Cases: Applying model to MCDMP Documentation Workflows

A process is as good as its functioning. We have drafted the process and believe it works, as is proven by several open source projects and companies such as Google, Twitter and Microsoft that have adopted similar agile documentation methodologies. In this section, we list potential use cases for this process, to begin with. The process must address these satisfactorily and needs to evolve as more use cases for documentation arise in future.



Use cases: Shared doc dev Process

- Fill in appropriate content in MCDMP Skeleton docs
- Add API documentation pertaining to a new MCDMP service
- Update API documentation pertaining to a new MCDMP service
- Browse existing documentation and contribute fixes, updates.
- User of MCDMP documentation finds issues, how is it reported, fixed, tracked and how does user know if it is fixed?
- How can product owners track critical documentation issues at any point in time?
- How to selectively publish public API info but not the private API one?
- Early user feedback, what works, what doesn't

Fig. 1.7: Figure: Architecture: Shared Document Development Model for MCDMP

1.7 Feedback on how to improve

Have you tried contributing to TechDocs yet? If not, I'd encourage you to do the same and give the process a shot. If something needs tweaking or can be made easier, better towards better scalability, document legibility or understanding, feel free to send your suggestions to TechDocs Support.

CHAPTER

TWO

LOOM DOC STYLE GUIDE

How do we make it really easy and simple for Loom users to adopt the platform?

The very basic ingredient is the Platform itself. It needs to be flexible, scalable, intuitive, easy to deploy and quick to gather information insights. Besides that, one of the key enablers towards facilitating onboarding is Loom user documentation. Documentation that satisfies the following criteria:

- Concise,
- Current & Continuously updated,
- Consistent,
- Clear, and
- Comprehensive

Loom follows a shared, scalable documentation model whereby several contributors work towards creating documentation that helps Loom user onboarding. In order to allow for more consistency across developer documentation, use this style guide.

If you are not aware of the Loom process for shared development of documentation, you may want to refer to *MCDMP Shared development of Documentation* before you read this style guide.

REST API Platforms such as Loom are designed to be consumed by APIs! It is important that we ensure Loom clients, or consumers, are able to implement these APIs and understand what is happening, with ease and swiftly. As we build Loom, we want to ensure that we not only provide information on these APIs to help developers integrate / debug connections and solutions around Loom, but also return back relevant data whenever a user makes a call, especially when the API call fails.

A big proportion of Loom Technical documents will come from and will be consumed by developers. The Loom Documentation process is geared for the developers. You can say, it is documentation of the developers, for the developers and by the developers. The ultimate goal is to make it really easy for developers, who don't know what our Loom product can do, or are not sold on using it yet, to get engaged and use it faster.

This guide is inspired by Google Style Guide which is adopted by several open source projects out there. Besides that, there are other references such as API Documentation Guide, and Best REST API Template. These are in use by many projects and many engineers, who collaborate globally to create world class software and documentation, used across verticals and industries.

2.1 Goals and Audience

The main purpose of this style guide is to clearly list and record simple guidelines that Loom team decides to follow as documentation style during shared document development process.

The guide can help developers, authors, technical writers to form a consistent approach for writing API documentation.

The primary goal of this guide is to codify and record decisions that Google's Developer Relations group makes about style. The guide can help you avoid making decisions about the same issue over and over, can provide editorial assistance on structuring and writing your documentation, and can help you keep your documentation consistent with our other documentation.

This guide is not intended to be an industry standard, nor does it apply to all Veritas documentation. This is valid only for Loom related platform and associated solutions.

2.2 How to use this Guide

This guide is a reference document. You can choose to read it linearly or look up specific issue or topic using the search box.

For issues not covered in this guide, see Google Style Guide or Other Style Guides.

2.3 Restructured Text Style Guide

You may also want to refer to some of the 'coding' standards applied to documentation files developed using Sphinx automation engine and Restructured Text formal. See Sphinx Documentation Style Guide for details.

2.4 Documentation Do's

Purpose	Guidelines
Clarity	Most of the principles that apply to good technical documentation also apply to accessible technical documentation: • Use correct grammar and punctuation. • Use active voice and present tense. • Write clear, simple sentences. • Be consistent.
Comprehension	Include screenshots! Users find screenshots very helpful, particularly annotated screenshots. When using screenshots as figures, remember to circle the relevant buttons, add arrows pointing out mentioned links, or highlight key sections.
Common norms	Spell out numbers nine and under. Use numerals for 10 and up.
Common norms	Use comma before "and" in a list of several items. For example: "This theme is elegant, simple, and easy to use."
Loom SDK | If you are documenting a workflow that uses Loom Platform SDK APIs, make sure Consistency | you include working code samples.	
Shared doc development norm	Do not delete, move or rename any document, page or topic which you do not own or did not create. Discuss and review as there may be cascading impact and changes required to be done for the same. Collaborate on such wide impact documentation changes.
Cross referencing common norm	Be aware that the content of some pages is included in other pages. Make sure you put your content in the right place, if it is cross referenced multiple times and also across guides. Follow similar principles just as in source code common files and function norms. This applies in particular to the Concepts and Usage Guide, FAQ, SDK API documentation.
Loom Document Consistency	If you are inserting a topic in an already existing piece of content, say a page or in a new section, ensure that you follow a consistent style, layout, grammar and format with the rest of the page.
Loom Version & Accuracy	If your update applies to a specific version of a product or framework add or update a panel at the top of the page: • Include the product version for which your update is relevant. • Use 'Available', 'Changed' and 'Deprecated'. Also, you could use the note directive to highlight the section you have updated and mention the relevant product version to which the change applies. {TBD: Example of the above}
Work in Progress Indicator	In a CI/CD shared documentation development model, it is important to let others know if your document is workable, usable but not yet reviewed, or ready to be released to the outside world. In other cases, it could be a piece of work in progress. When you are writing a lengthy piece of documentation, you may need to publish it for internal users to start integrating it, even

2.5 Documentation Do not's

Purpose	Guidelines
No Cryptics	No Loom Buzzwords please! Before you use any acronym, expand it for first time use, with acronym in brackets. For subsequent use, acronym is fine. **Not recommended**: Use JBMgmtSvc to schedule this task if this policy is breached. **Recommeded**: Use Job Management Service (JBMgmtSvc) to schedule this task if privacy policy is breached. Refer to JBMgmtSvc details here (refer to link) on how to set up job management schedule
No Complicated Language	No long sentences. Think of the twitter 140 characters generation. Avoid choppy sentences. **Tip:** Think of your audience **Compare the first paragraph below to the second in terms of complexit** • When you write, imagine you speaking these very same words as if you were saying these to someone in a conference call or Webinar where no real real time clarification or feedback is possible by those who'd read it and try to understand what you wrote at a later point in time. • When you write, imagine your audience reading what you write, at a later point in time. Face to face conversation allow the luxury of real-time clarifications, additional cues via facial expressions and tone. Written word is more powerful and could be confusing as it lacks that luxury.
Factually Correct	Avoid using 'as of now', 'at this time'. Evolving platforms such as Loom will keep changing frequently to meet user needs. Whatever you write, must be factually correct for the corresponding Loom features available to users.
Avoid Repetition	Use of repetitive phrases such as 'To do', or 'Here is', or "You can" at the start of each sentence is detrimental to reader attention.
Enterprise Focus	Avoid current pop-culture references.
Appropriate Language	No Jokes at the expense of customers, competitors or anyone else for that matter.
Spoiler Alert	Avoid trying to document or talk about future features or solutions or products or enhancements, even in innocuous ways.
Ambiguity	Avoid the modal verbs such as could, may, might, and should in technical documentation. Modal verbs are ambiguous and leave the reader wondering what to do. For details, visit Modal Style Guide.

CHAPTER

THREE

LOOM DOC TEMPLATES

3.1 Template: Loom Service FAQ

3.1.1 Loom Service Name

Update the Loom service name in the title above. For e.g., XYZ Service.

What is XYZ Service?

Describe the Loom service in terms of what it does. How does a service user benefit from it?

Service Dependencies

If there are any other Loom micro-services that you need to deploy in order to use this service, then list those. Point to those service FAQs if available.

When to use this service?

In brief, describe the context when this service is used. For example, job management service is used in the context of “information classification”, whereby some long running tasks and subtasks need to be managed by the job manager service.

Service Usage REST API Workflow

Give a high level snapshot of how a developer (user) can utilize this service and what is a typical API workflows?

Service FAQ Section

This section lists Q and A relate to this service.

You may want to refer to the srv_idm_faq as an example.

The following question and answers are listed as an example. They may not be relevant to your service. As you reuse this template, replace them with actual ones for your service.

What API do I need to call in order to list all the components that are part of Loom control plane, data plane?

Answer TBD

Is there an API provided by Cluster Manager Service that returns health status of a cluster?

Use clustermgr/v1/cluster/{clustername}/health. Note, this returns health of a specific cluster.

If a developer (user) queries an object, is it possible to obtain cluster health as part of JSON query?

Yes. (This may need to be reviewed and reworded properly in the context when such as need arises.)

Is there an API that provides list of all services and nodes corresponding to a given Loom component?

Use clustermgr/v1/health API to obtain information about operational analytics of VMs and overall cluster.

Is there an API to obtain various links between Loom components?

No API's defined for that.

3.2 Template: Loom XYZ Service Troubleshooting Guide

Warning: This is a sample template for Loom Troubleshooting Guide. This document is under review and subject to change during Loom Alpha release timeframe.

3.2.1 Service Issue Lorem Ipsum

Description Explain the issue that a Loom Service User might face.

Another para describing the issue in detail. You could also include screenshots or a flowchart highlighting the steps that cause this issue.

Workaround Describe workaround to address the issue listed above.

Another para that may list details of the workaround.

This is last para of the workaround. You can add screenshots here too. For example, refer to the picture below to highlight some section or steps.

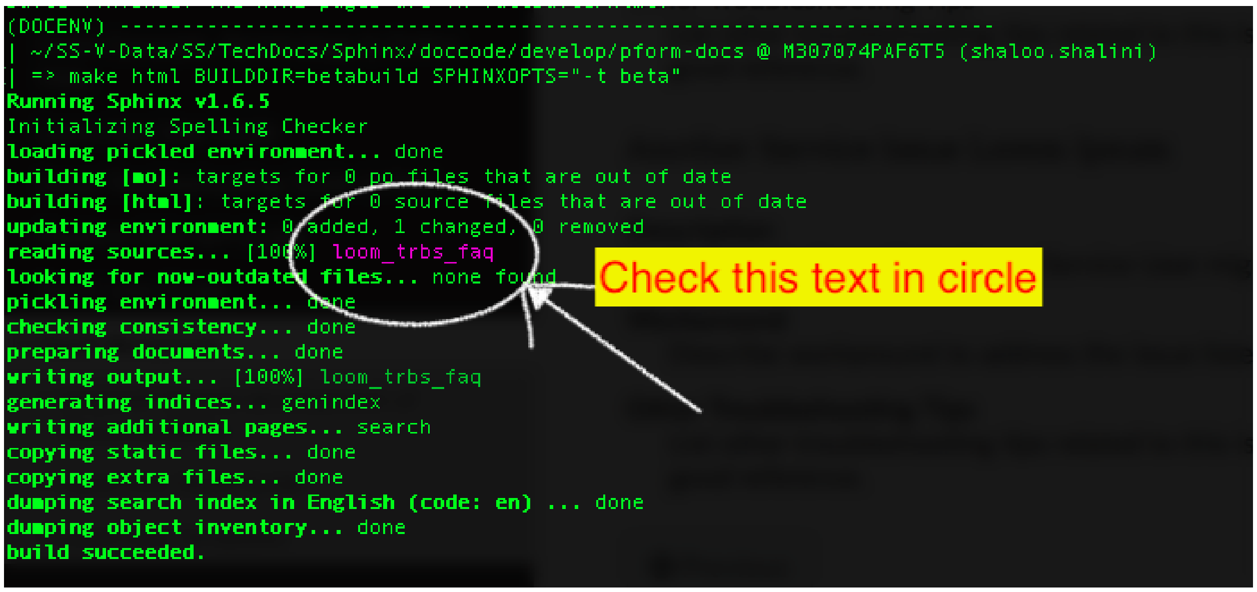
Other Troubleshooting Tips List other troubleshooting tips related to this issue or point to other similar issues that may be a good reference.

3.2.2 Another Service Issue Lorem Ipsum

Description Explain the issue that a Loom Service User might face.

Workaround Describe workaround to address the issue listed above.

Other Troubleshooting Tips List other troubleshooting tips related to this issue or point to other similar issues that may be a good reference.



```
(DOCENV) ------------------------------------------------------------------
| ~/SS-V-Data/SS/TechDocs/Sphinx/doccode/develop/pform-docs @ M307074PAF6T5 (shaloo.shalini)
| => make html BUILDDIR=betabuild SPHINXOPTS="-t beta"
Running Sphinx v1.6.5
Initializing Spelling Checker
loading pickled environment... done
building [mo]: targets for 0 po files that are out of date
building [html]: targets for 0 source files that are out of date
updating environment: 0 added, 1 changed, 0 removed
reading sources... [100%] loom_trbs_faq
looking for now-outdated files... none found
pickling environment... done
checking consistency... done
preparing documents... done
writing output... [100%] loom_trbs_faq
generating indices... genindex
writing additional pages... search
copying static files... done
copying extra files... done
dumping search index in English (code: en) ... done
dumping object inventory... done
build succeeded.
```

Fig. 3.1: Figure: Troubleshooting Image Sample