LFN Project Data Template

Instructions

This document is a template to collect the data LFN thinks are relevant to LFN Project health. Projects will copy this template and instantiate it with their data as a part of applying for lifecycle state transitions or induction into the LFN.

> *Click the "..." in the upper right hand corner of this page. Click "Copy". Choose "Project Proposals". Rename the title to your Project*

If the project has instantiated this template before, they are welcome to start from that base and update it with any new information. Be sure that the data requested by the primary copy of the Project Data Template has not changed.

These instructions should be removed from the instantiated template.

Project Data

LFN Data Template.pptx

Below are additional details about how to fill out the above presentation:

Project Vitals

This slide is an overview of the fundamental information about your project.

- Project Name: Please fill in the official name of the project. This is the name that will be used in all official documentation and communication.
- Project Creation Date: Enter the date when the project was officially started or initiated.
- Project License: Specify the license under which your project is distributed. Examples include Apache 2.0, MIT, GPL, etc. If you have multiple licenses, please list them all.
- Contributing Organizations: Here, you should list all organizations that are actively contributing to the project. This includes companies, academic institutions, non-profits, etc.
- Seed Code Availability: Enter the date when the seed code, or initial codebase, of your project will be or was made available. Also, list the repositories where this code can be found.
- Release Schedule: This is where you detail the timeline of your project's release. It might include dates for the start of development, feature freeze, code freeze, testing periods, and the final release date. If you have a consistent release cycle (e.g., every 6 months), you can specify that here.

Project & Community Resources

This slide provides information about the resources that your project has in place to foster communication, collaboration, and transparency in your community.

- Technical Charter: Please provide a link to the technical charter for your project, and also note the date it was established. The technical charter outlines the mission, scope, and operations of the project.

- TSC Established: Note the date when the Technical Steering Committee (TSC) was established for the project. The TSC typically oversees the technical direction of the project.
- Website: Provide a link to the project's official website, if it exists. This is where newcomers can find general information about the project.
- Wiki: If there is a project Wiki, provide the link here. This can be a valuable resource for detailed technical information, community guidelines, and more.
- Mailing list: Include a link to the project's mailing list, which is a key communication channel for many open source projects.
- Community size (based on mailing list): Indicate the size of your project's community. A simple way to estimate this is by looking at the number of subscribers to your project's mailing list.
- Community Meetings: Provide information about any regular meetings held by your community. This can include the meeting schedule, agenda, and how to participate (e.g., a Zoom link, IRC channel, etc.).

Project Governance

This slide offers information about how your project is organized and managed, and how decisions are made.

- Project Roles: Briefly describe the various roles within your project (e.g., Project Lead, Maintainer, Contributor, etc.) and what responsibilities each role carries.
- How people come to fill project roles: Outline the process by which individuals are chosen or elected to fill specific roles within your project. This might involve a vote by existing members, an application process, or some other method.
- How people are removed from project roles: Detail the procedure for removing individuals from their roles, whether due to inactivity, misconduct, or some other reason.
- Who currently fills all project roles: Provide a list of the individuals who currently occupy each role within your project.
- How disputes are definitively resolved: Explain the system for resolving disputes within your project. Typically, this is done by a majority vote among certain project members.
- How governance evolves over time: Describe how your project's governance structure may change and adapt over time. This might involve the creation of new roles or the modification of existing ones.
- What is the top-level technical decision-making body for the project, analogous to a TSC, to which the TAC should look for interfaces? Provide information about the primary technical decision-making body of your project. This is typically a Technical Steering Committee (TSC) or something similar.

Remember, this slide should provide a clear overview of how your project's governance operates. The clearer this is, the easier it will be for newcomers to understand how they can participate and contribute to the project.

Use Case / Problem Statement

This slide is your opportunity to clearly define the specific problem your project aims to solve, as well as the use case scenarios your project addresses. This will help the TAC understand the context and importance of your project. Here are some pointers to guide you through the process:

1. Problem Statement: This is where you define the problem that your project is designed to solve. Be specific about what the problem is, who it affects, and what implications it has. The problem statement should be succinct, focused, and clear.
2. Use Case Scenarios: List and describe typical situations where your project would be applied. Who are the users? What are their goals? What tasks are they performing, and how does your project assist in accomplishing these tasks? Be sure to highlight how your project is unique in addressing these needs.

Remember, clarity is key. Provide enough detail to comprehensively view the problem and use case scenarios, but strive to keep your language simple and direct.

Project Architecture

This slide is dedicated to presenting your project's architecture. Your aim here is to give the TAC a clear understanding of how your project is structured and how its various components interact. Below are some guidelines to assist you:

1. High-Level Overview: Provide a general description of your project's architecture. This can include the guiding principles behind its design, the architectural style or pattern it follows (e.g., microservices, event-driven, layered), and any notable characteristics.
2. Component Diagram: Include a diagram illustrating the main components of your project and their interactions. This visual representation can greatly enhance understanding. Be sure to label all parts of the diagram and provide a brief description of each component's function.
3. Key Interfaces & Dependencies: Describe the critical interfaces within your project and any external dependencies.
4. Data Flow: If applicable, briefly explain how data flows within your system. This can be included in your component diagram or described separately.
5. Future Evolution: Briefly outline any planned or potential changes to your project's architecture.

Remember, your goal here is to provide a clear, concise overview that's understandable even to those who may not be familiar with every technical detail.

Infrastructure Tooling

In this slide, you are expected to present an overview of the tools your project utilizes for its infrastructure. These are crucial to your project's operation, allowing your team to collaborate effectively, track progress, handle issues, and maintain high-quality standards.

When filling out this slide, please provide the following details:

1. Wiki: Share a link to your project's wiki page, if applicable. This is where you can house essential project documentation, updates, and information.
2. Code Repositories: Include a link to where your code is hosted. This could be a platform like GitHub, GitLab, or BitBucket.
3. Bug Tracking: Specify the tool you're using to track bugs and issues, such as JIRA, GitHub Issues, or Bugzilla. Include a link to your project's specific bug tracking page or system.
4. Code Review: Indicate what system or tool you're using for code reviews. This could be Gerrit, GitHub PRs, etc. Don't forget to provide a link to where these reviews take place.

5. Documentation: Share where the project's technical and user documentation is stored and accessible. This could be on a dedicated website, within the code repository, or elsewhere.
6. CI/CD Tooling: Specify the continuous integration/continuous deployment (CI/CD) tooling you use. This could be Jenkins, GitLab CI, GitHub Actions, etc.
7. Collaboration tools: Mention any tools you use for project communication and collaboration. This could include mailing lists, forums, chat platforms (e.g., Slack, Teams), video conferencing tools, etc.

Remember, it's not just about listing the tools—you should provide direct links and briefly explain how each tool contributes to your project's infrastructure and workflow.

Roadmap

In this final slide, you are to provide an overview of your project's future trajectory. The roadmap should highlight the key milestones and deliverables your project aims to achieve in the upcoming quarters or years. Here's a guide to help you fill it out:

1. Near-Term Objectives: List your immediate goals for the project. What are the critical tasks or features that you aim to complete in the next few months? Ensure these objectives are clearly defined, achievable, and align with the overall project vision.
2. Long-Term Objectives: Detail the broader, strategic goals for your project. Where do you see your project heading in a year or more? Long-term objectives should be ambitious, yet attainable, driving the project towards continuous growth and innovation.
3. Key Milestones: Highlight the significant events or checkpoints in your project timeline. These could include anticipated release dates, feature completion, important meetings, or any other noteworthy occurrences.
4. Risks & Challenges: Identify potential obstacles that could impact the roadmap. Discussing these openly helps to prepare the team and stakeholders for any contingency and fosters a proactive approach to problem-solving.
5. Timeline: Provide a visual representation or description of your roadmap's timeline. This will give your audience a clear understanding of your project's path and the order in which you expect to achieve your objectives.

Remember, the roadmap is a strategic document that guides your project. It's okay if it changes over time; the aim is to provide a clear, ambitious, yet realistic, direction for your project's future.

Vocabulary Reference

Explanations of domain-specific vocabulary.

```
.. todo:: Look into using special rst to make these definitions into tooltips
```

```
.. todo:: Consider extracting this to a stand-along file so can reuse elsewhere
```

- Active
  - In this context, typically related to the activity level of a project or person.
  - As a person: "Foo Committer on Bar Project has not sent any patches or done any code review for Bar in the last 12 months. Bar's Project Lead reached out to Foo Committer to discuss transitioning to an Emeritus Committer."
  - As a project: "Bar Project has not had any non-trivial code changes merged in the last 12 months. The LFN TAC reached out to Bar Project to discuss transitioning to the LFN Archived lifecycle state."
  - The LFN norm for "active" is about 12 months.
- Committer
  - Person with permission to cause commits to be merged into a project's source control repositories.
- Contributor
  - Person who has contributed to a project. "Contributions" are broadly defined. Examples include things like code, documentation, and bug tracker changes.
- Diverse
  - In this context, typically related to the number of different organizations involved in a project.
- Downstream
  - In this context, typically means the products based on a project. Community collaborates on upstream project, which is downstreamed by a company into a product.
  - Alternatively, could relate to a relationship between two "upstream" open source projects (not by-company products) where one consumes (is downstream of) the other.
  - As a verb: "to copy something from the open source project to a product based on it".
  - As a dependency relationship: "Linux is a downstream of C".
- Upstream
  - In this context, typically means the main open source project a community collaborates on. The code, tooling and people that comprise a project.
  - As a verb: "to add something to the main open source project".
  - As a dependency relationship: "C is an upstream of Linux".