Technical Work in the EdgeX Foundry Project

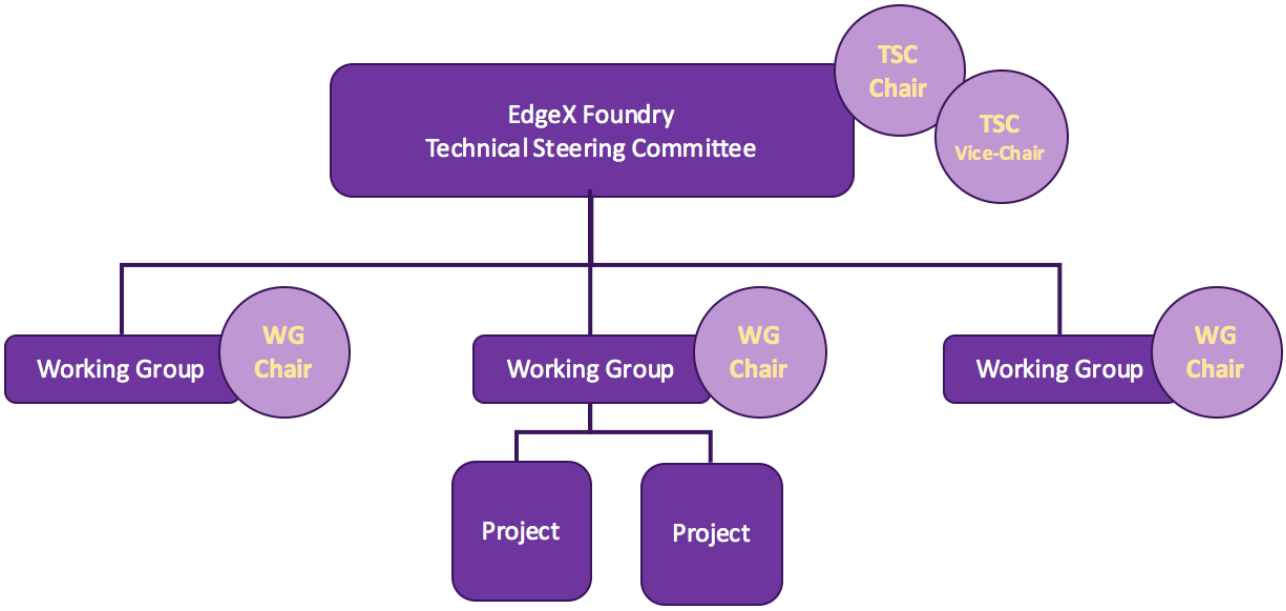
- Technical Steering Committee
- Working Groups
- Projects
  - Project Leadership Team and Project Lead
- Making Contributions
- Release Process
- Decision Making, Conflict Resolution, Role Nominations and Elections
  - Review Then Commit
  - Expressing Agreement and Disagreement
  - Lazy Consensus / Lazy Voting
  - Informal Votes or Surveys
  - Leadership Team Decisions
  - Conflict Resolution
- Acknowledgements

Updated 5/31/23 - per voted TSC changes during Napa Pre-Wire meeting

Technical Steering Committee

The TSC is a committee composed of technical leaders from the open source project responsible for oversight of the technical codebase, the technical community and release process.

The TSC operates in the context of the Technical Charter by these principals within this Code of Conduct.



The TSC is led by a **Chair** elected per the EdgeX Foundry Technical Charter.

The TSC Chair is responsible for acting as a liaison between the Governing Board and technical leadership of the Project, coordinating and leading the recurring TSC meetings and managing the overall TSC structure and work progress.

The TSC is comprised of the TSC Chair, each of the Working Group Chairs, and other members as specified in the EdgeX Foundry Technical Charter.

Each Working Group must have a chair and optionally a co-chair (or vice-chair). The Co-chair role assists in working group leadership and serves as a proxy for the chair when the work group chairperson is not present/available. While not official, the co-chair serves as heir apparent for the chair role for future project efforts. Each Working Group has a single vote in TSC matters - it can be either the chair or co-chair (i.e. - only one vote for TSC matters is given to each work group) .

Working Groups

The TSC is comprised of **Working Groups**.

> The term “**Working Group**” (WG) within the EdgeX Foundry Project will refer to an organizational unit of the Technical Steering Committee (TSC) focused on a specific technical domain e.g. Device Service Working Group.

Working Group Leadership

Each Working Group is led by a Working Group Chair who is elected from among project leadership team members of the projects in the Working Group. The Working Group Chair is responsible for coordinating the efforts of the various projects chartered in the Working Group.

Projects

Each Working Group has one or more **Projects** under it. The Working Group coordinates the efforts of these projects from an architectural, technical and organizational perspective to ensure that the work done by these projects is unified, consistent and aligned with the architecture and goals of the Working Group and the larger EdgeX Foundry Project.

> The term “project” within the EdgeX Foundry Project will refer to a collaborative endeavor to deliver a work item.

A “Technical Project” is a collection of specific Maintainers, users and other developers, code, releases, issues, and other activity oriented around a common software-implemented mission. A Technical Project may be to develop a new capability or to refactor or remove an existing capability for the EdgeX technology releases. Such projects may take the form of a new component or may propose additions, deletions or changes to an existing component.

A 'Functional Project' is intended to produce a document, such as a requirements or use cases document, a white paper, or analysis and there are 'Technical' projects.

All projects have a page on the EdgeX Foundry Wiki.

New Projects are proposed using the Project Proposal Template.

Projects progress through a standard Project Lifecycle.

Project Leadership Team and Project Lead

Projects in the EdgeX Foundry Project are managed by a Project Leadership Team. The leadership team is made up of distinguished community members, but the exact composition may depend on the project. The leadership team owns the projects processes, the overall architecture and all assets within the project and makes sub-project wide decisions on behalf of its community.

Project Leaders coordinate with the Chair of their respective working group through regular Working Group meetings to coordinate the efforts of all the projects that are chartered under a working group.

A projects leadership team members are listed on the sub-project's team portal (i.e. EdgeX Foundry Wiki).

The Leadership Team may elect a Project Lead who is also a member of the Leadership Team. Project Leads are the public figurehead of the project and are responsible for the health of the project. Project Leads can also act as referees should the Project Leadership Team become paralyzed.

Making Contributions

The EdgeX Foundry Project does not require community members to sign contribution or committer agreements. We do require contributors to sign contributions using the sign-off feature of the code repository, following the same approach as the Linux Kernel does (see Developer Certificate Of Origin).

More information on making contributions can be found in the following documents:

- Contribution Guidelines
- Review Then Commit Policy

Release Process

Code is released under the following Release Taxonomy.

Decision Making, Conflict Resolution, Role Nominations and Elections

EdgeX Foundry Working groups and their projects are normally auto-governing and driven by the people who volunteer for the job. This functions well for most cases. This section lists the main mechanisms by which projects make decisions. This section lists the default mode of operation. Projects can deviate from the default, but are required to document deviations from the default and link to it from this XEN document. The only exception is that each project is required to adhere to the **Review Then Commit Policy**, **Leadership Team Decisions** and **Conflict Resolution**.

Review Then Commit

The vast majority of technical decisions within the EdgeX Foundry Technical Projects are code related decisions, which determine whether a specific change can be accepted into the code base. The default decision making process is a review and commit process, which requires that all changes receive explicit approval from respective code owners (maintainers) before they are committed. The exact workflow and details of this policy between sub-projects may differ and are documented in one or several of the following places: MAINTAINERS/README/CONTRIBUTING files in repositories and/or the sub-project team portal.

Expressing Agreement and Disagreement

Within the community, we follow the following number notation to explicitly express opinions on proposals, formal or informal votes.

- **+2** : I am happy with this proposal, and I will argue for it
- **+1** : I am happy with this proposal, but will not argue for it
- **0** : I have no opinion
- **-1** : I am not happy with this proposal, but will not argue against it
- **-2** : I am not happy with this proposal, and I will argue against it

A **-2** should include an alternative proposal or a detailed explanation of the reasons for the negative opinion. A **+2** should include reasons for the positive opinion.

How we tally results and their implications depend on the context in which is is used and are marked with Passed/Failed: in one of the following sections:

- Lazy Consensus / Lazy Voting
- Leadership Team Decisions
- Project Wide Decision Making

Lazy Consensus / Lazy Voting

Lazy Consensus is a useful technique to make decisions for specific proposals which are not covered by the Review Then Commit Policy or do not require a more formal decision (see below). Lazy Consensus is extremely useful, when you don't anticipate any objections, or to gauge whether there are objections to a proposal. The concrete process in this section is a mixture between Lazy Consensus and Lazy Voting and is designed to avoid unnecessary multiple stages in decision making.

To make use of it, post something like the following on the project's mailing list (or some other communication channel):

```
I am assuming we are agreed on X and am going to assume lazy consensus:
if there are no objections within the next seven days.
```

You should however ensure that all relevant stake-holders which may object are explicitly CC'ed, such as relevant maintainers or committers, ensure that **lazy consensus** is in the body of your message (this helps set up mail filters) and choose a reasonable time-frame. If it is unclear who the relevant stake-holders are, the project leadership can nominate a group of stake-holders to decide, or may choose to own the decision collectively and resolve it.

Objections by stake-holders should be expressed using the conventions above to make disagreements easily identifiable.

Passed/Failed: The proposer of Lazy Consensus decision is assumed to implicitly have an opinion of **+1**, unless otherwise stated.

- Failed: A single **-2** by a stake-holder whose approval is necessary
- Failed: A total sum of opinions **<=0**
- Passed: A total sum of opinions **>0**

It can only be overturned if the project leadership agrees collectively, that the decision is too important to be settled by lazy consensus / lazy voting. In situations where a proposal is failed, an alternative solution needs to be found, or if a decision is formally challenged, conflict resolution mechanisms may need to be used to resolve the situation.

Further Examples: A Lazy Consensus decision starts out with the implicit **+1** opinion of the proposer. If there is no explicit response, the proposal passes as the sum is **>0**.

If there is a single **-1** without any **+** votes, the proposal fails.

If there are multiple **+1**'s or **+2**'s, more **-1**'s than positive votes are needed for the proposal to fail. This mechanism, is often also called **Lazy Voting**.

The process does allow for a proposer to state a starting opinion of **0** or **-1**. In this case, the Lazy Consensus label does not work for the process, as positive opinions are needed for the proposal to pass. To make use of this mechanism, post something like the following on the project's mailing list (or some other communication channel)

```
I want to solicit opinions on X and am going to assume lazy voting:
My starting position is **0**, as I feel that at least one other
stake-holder should agree with the proposal.
If there is a majority in favour, without a **-2** objection within the
next seven days, I assume that the proposal holds and does not need
require further discussion.
```

Unlike in the lazy consensus case, a single **+1** vote is needed. Otherwise the proposal fails. Otherwise, the counting rules follow the general case.

This can be useful in situations, where the proposer is not quite sure about his/her position, or where the invoker acts on behalf of the community to resolve a discussion which has become stuck. A starting position of **-1** can be used to verify that a specific approach may be a bad idea: whether this is really useful, has to be verified as we start using this process.

Informal Votes or Surveys

Generally the EdgeX Foundry community tries to achieve consensus on most issues. In situations where several concrete options are possible, community members may organize an informal vote on the different proposals and use the conventions above to identify the strongest proposal. Once the strongest candidate has been identified, lazy consensus could be used to close the discussion. In some situation, a specific survey may need to be created, to help identify gauging consensus on specific issues. For informal votes and surveys, we do not prescribe specific rules, as they are non-binding: it is up to the organizer of an informal vote or survey to interpret the result and explain it to the community. If the vote/survey relates to an area that is owned by the project leadership, the project leadership has to formally confirm the decision.

Note that informal votes amongst a small set of stake-holders that disagree on a position during technical disagreements in code, design reviews and other discussions can be useful. In technical discussions it is not always clear how strong agreement or disagreement on a specific issue is. Using the conventions above, can help differentiate between minor and major disagreements and reduce the time a discussions continues unnecessarily. This is true in particular for cases, where several maintainers may need to agree to a proposal.

When having an informal vote or survey, they creator should consider whether conducting a vote or survey in public, may be divisive and damaging for the community. In such cases, the vote/survey should be conducted anonymously.

Leadership Team Decisions

Each project has a leadership team, which is typically made up of the most senior and influential developers within the sub-project (e.g. the project's committers). The project leadership team owns decisions, such as:

- Project wide policy decisions (e.g. policies, procedures and processes whose scope is specific to the sub-projects). This includes deviations from project global governance, where permissible.
- Decisions related to project assets that are not clearly owned (e.g. unowned code, project wide assets such as test infrastructure, etc.).
- Decisions related to nominating and confirming leadership roles within the sub-project. This includes decisions to creating and filling specialised new roles, such as release managers or similar, including their scope and set of responsibilities.
- Resolving conflicts within the sub-project that cannot otherwise be resolved.

Leadership team decisions can be made in private (e.g. a private IRC meeting, on a private mailing list, through a private vote) or on a public mailing list using decision making conventions. If a decision is made in private, the outcome must be summarized in terms of number of votes in favor or against on a public mailing list. Decisions should **not** generally be made in an anonymous vote, unless there is a good reason to do so. For example, if the decision may be divisive and damage the cohesion of the leadership team, an anonymous vote is preferred. In such cases, the leadership team, can ask the community manager, to arrange an anonymous vote on behalf of the leadership team.

Decisions (also called Resolutions) require a **2/3rd** majority amongst active leadership team members in favour of a proposal. The tallying of votes follows the rules outlined below. Note that a minimum of 3 leadership team members is needed for a leadership team to function.

Leadership team decisions normally have to be made actively: in other words each team member has to cast a vote **explicitly** expressing their opinion. The only exception are face-2-face or on-line meetings with a quorum of **2/3rd** of active leadership team members present at the meeting: in such cases a meeting chair is required who calls for decision on a resolution and asks for objections. This allows to conduct meetings more quickly.

Passed/Failed Resolutions:

Voting is conducted in line with the following rules:

- Project leadership team members vote for (**+1**) or against (**-1**) a resolution. There is no differentiation between **+1**/ **+2** and **-1**/**-2**: in other words a **+2** is counted as a vote for, a **-2** as a vote against the resolution. The number of votes for and against a resolution is called **active vote**. **0** votes **are not counted** as an active vote.
- A **quorum of at least 1/3 of positive votes for a proposal** is required for a resolution to pass. In other words, if the leadership team has 7 members, at least 3 members need to vote for the resolution.
- The resolution passes, if a 2/3 majority of active votes is in favour of it.

The table below maps the number of leadership team members against the required quorum:

Leadership team members	10	9	8	7	6	5	4	3	2
Positive votes needed for quorum	4	3	3	3	2	2	2	1	1

The table below maps active votes against votes needed to pass:

Active Votes (+1 or -1)	10	9	8	7	6	5	4	3	2
Positive votes needed to pass	7	6	6	5	4	4	3	2	2

Conflict Resolution

Projects in EdgeX Foundry are not democracies but meritocracies. In situations where there is disagreement on issues related to the day-to-day running of the project, the project leadership team is expected to act as referee and make a decision on behalf of the community. Projects leadership teams can choose to delegate entire classes of conflict resolution issues to community members and/or the project lead (e.g. the project can choose to delegate refereeing on committer disagreements to the project lead; or it could choose a specific committer to always act as referee amongst a group of committers). Any such delegation needs to be approved as normal and has to be documented.

Should a project leadership team become dysfunctional or paralyzed, the project leadership team or project lead should work with their Working Group Chair or the TSC to find a way forward.

In situations where the entire EdgeX Foundry community becomes paralyzed the impacted working groups and project leadership teams should work with the TSC to find a way forward.

Acknowledgements

The above leverages process utilized by the Xen Project