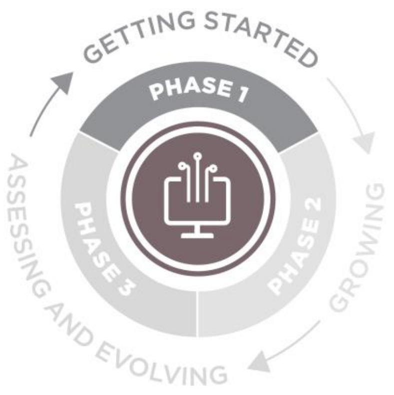


TECHNOLOGY

Phase I: Laying the Groundwork

Activity 1: Who Are Your Stakeholders?

Goals

1. Identify and prioritize the community's technology stakeholders

- A stakeholder is any individual, group, or organization within or outside the project who is impacted by its outcome or who has an interest in its success

2. Create a high-level map of the technology stakeholder's characteristics
3. Identify the program's goals for high-priority stakeholder groups

Prerequisites

CE Activity: Who Is Your Community?

Who Should Participate?

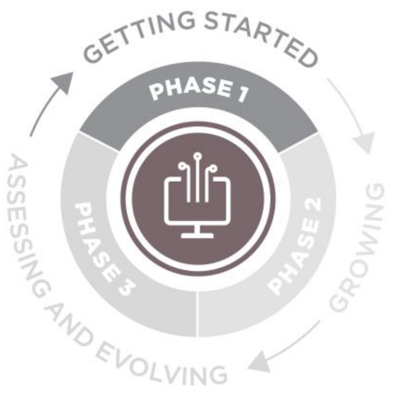
Program leadership (strategic thinkers), Program management (tactical thinkers), Program staff (operational experience)

Length

60-90 minutes

Activity Instructions

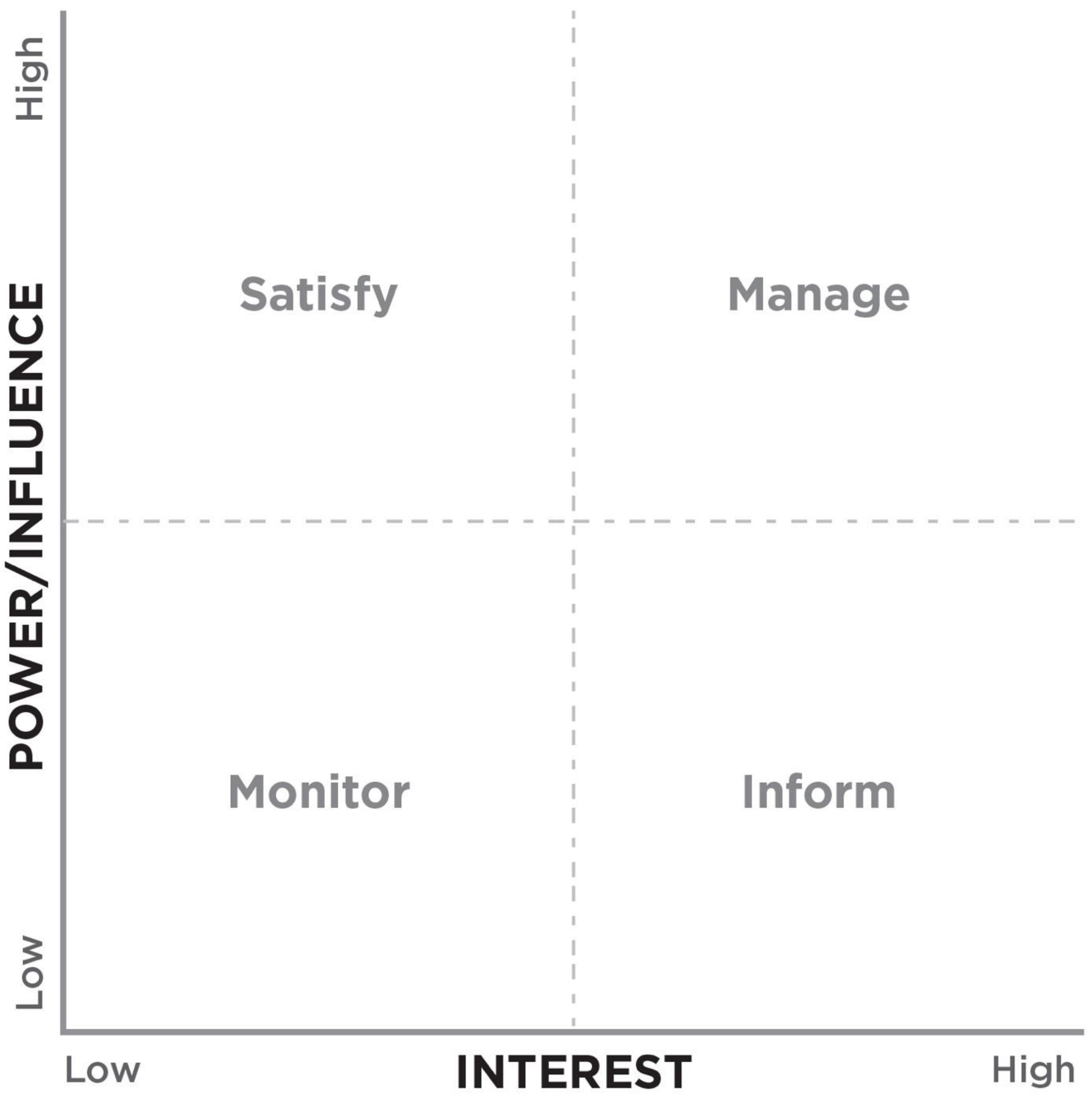
1. On a whiteboard or shared online document, identify your potential technology stakeholder groups. Using sticky notes (real or virtual) works best, as ideas will be moved around in later steps.
    - It's ok to be granular - your team will eliminate duplicates / combine stakeholders in the next step.
    - Stakeholder examples: end users, QA testers, code contributors, software engineers, sysadmins/IT staff at institutions using the software, program staff at related OSS programs, functional requirements contributors, etc.
2. Move the sticky notes around to categorize your stakeholders into groups.
    - Group examples: Technical contributors, Service providers, etc.
3. Select 3-5 groups to prioritize over the next year. An interest/influence grid or sticker vote may be helpful in prioritizing.
    - An interest/influence grid plots stakeholder groups against two axes: Interest and Power/Influence and then suggests a level of engagement based on their place on the grid. A sample grid is below.
    - In a sticker vote, each participant is assigned a number of stickers - these can be physical stickers in an in-person event or a specified piece of text (e.g. +1) in a virtual environment. Participants place their stickers or text alongside the options they're voting for, according to the parameters of the exercise.
4. For each prioritized stakeholder group in your diagram, discuss the following questions:
    - What are our goals for each user group or technology stakeholder group?
    - Are there shared or related goals across stakeholder groups? What are the opportunities and areas of collaboration? How can our community work together to create and achieve things?
5. If you have completed Technical Skills Inventory Part 1, you may also answer:
    - What technology skills are required to make our mission a reality? Do our prioritized stakeholder groups have these skills?

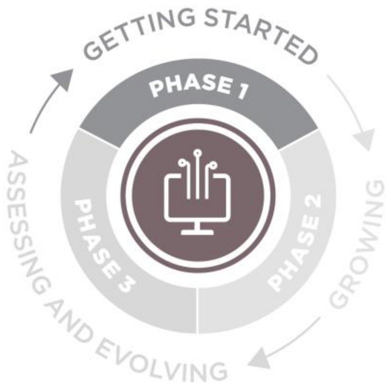


TECHNOLOGY

Phase I: Laying the Groundwork

Activity 1: Who Are Your Stakeholders?





TECHNOLOGY

Phase I: Laying the Groundwork

Activity 2: Technical Skills Inventory, (Part One)

Goals

1. Create an inventory of what technical skills are needed for technical staff, contributors, and users to develop, support, and maintain the platform.

Prerequisites

None

Who Should Participate?

Program management (tactical thinkers), Program staff (operational)

Length

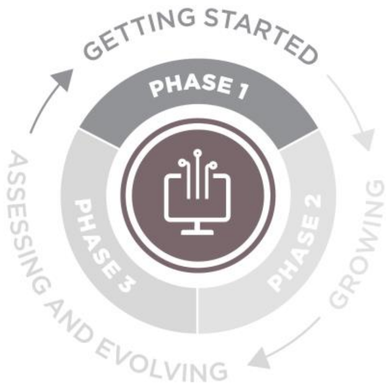
This activity does not need to be done as a group; it can be completed asynchronously/ collaboratively in a shared document.

Activity Instructions

1. Create a list of the technical roles held by program staff/stakeholders. Roles should be fairly granular, describing one aspect of developing, maintaining, or supporting your application.
2. Fill out the below template to add the technical skills required to fulfill the roles you listed. Examples in blue.
3. Once it's complete, sharing the skills inventory as part of your public documentation is a great way to let potential users and contributors know what types of skills are needed to take on certain tasks.

Once complete, this activity can be used in several additional contexts, including:

- Roadmap planning: ensuring that major deliverables on the roadmap are not all clustered around a certain skill set (and therefore a certain person).
- Community building: If users self-hosting is important to your program, the skills inventory can be used to compare skills required to install, upgrade, and maintain the software against skills that your end users have or have access to in their organizations.



Activity 2: Technical Skills Inventory, (Part Two)

Example roles (depending on tech stack, may need to qualify with frontend, backend, etc.)

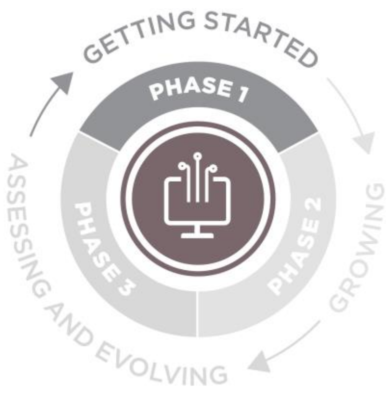
Technical lead, Code contributor, System administrator (e.g. install, upgrade), End user, Migration specialist, Technical support for End user

Example skills (include specific tools in table, e.g. HTML for Web design or development)

Web design, Web development, Assistive technology, Database, Data warehousing, Data analysis, GIS, Platform/OS, Quality assurance/Testing, Reporting, Security, Server

Sample Technical Skills Inventory – Examples in *Italics*

		Skill Level		
Role	Required skill	Novice	Intermediate	Advanced
Developer: Front end	*Angular/Typescript*	*X*		
	HTML		*X*	
	Bootstrap	*X*		
	SAAS/CSS		*X*	
System administrator (install, upgrade)	*Command line*		*X*	
	Package manager		*X*	
Report writer	*SQL*			*X*
	Crystal Reports			*X*
…	…			



TECHNOLOGY

Phase I: Laying the Groundwork

Activity 3: Personas and Pathways

Goals

1. Learn how to bring contributors onto your project by using tools called "personas" and "pathways"
2. Help your program plan and test how you'll interact with new contributors
3. Imagine what is really involved for new contributors to succeed

Prerequisites

Suggested: Who Are Your Tech Stakeholders and Technical Skills Inventory, Part 1.

Who Should Participate?

Program management (tactical thinkers), Program staff (operational expertise)

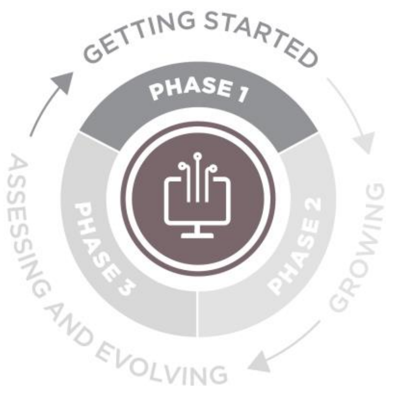
Length

60-90 minutes

Activity Instructions

Complete the Mozilla Open Leadership Activity: Contributor Personas and Pathways:

https://mozilla.github.io/open-leadership-training-series/articles/building-communities-of-contributors/bring-on-contributors-using-personas-and-pathways/



TECHNOLOGY

Phase I: Laying the Groundwork

Activity 4: Landscape Analysis

Goals

1. Understand where your program fits in the tech landscape
2. Use results to innovate, make decisions, identify opportunities for collaboration, increase usefulness/effectiveness

Prerequisites

Gov Activity: Mission/Vision

Activity: Who Are Your Technology Stakeholders

Who Should Participate?

Program management (tactical) with initial brainstorming input from program leadership and program staff.

Length

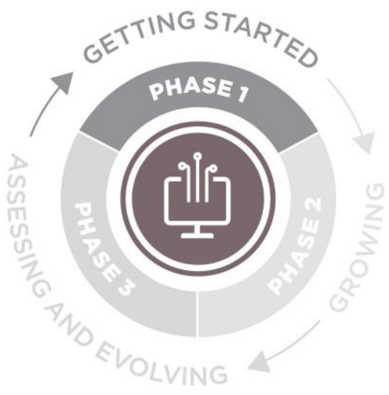
90 minutes

ACTIVITY INSTRUCTIONS

1. Build as comprehensive a list as possible of the competitive landscape - programs that are engaged in roughly the same work, serving roughly the same stakeholder groups. This list can be created during an in-person brainstorming session, or offline through a shared document or virtual whiteboard.
2. Create an assessment template that includes (at least) the following categories:
   a. Program name
   b. High level purpose
   c. Core features/functionality
   d. Target audiences
   e. Mission, values and vision — What are the specific, tangible goals they're trying to accomplish?
   f. Unique value proposition — What does the program claim to do that is different from other programs?
   g. OSS license (or note if it's a proprietary application)
3. Analyze the landscape by thinking through the following questions:
   a. How is your program different? How is it the same?
   b. What are you doing better? What can you highlight in your messaging and communications that is unique and will resonate with your target stakeholders?
   c. Where are your competitors excelling? What can you learn from them?
   d. Where are your competitors falling short? Are there any strategies that aren't working? Could you do them differently or better?
   e. What "gaps" do you see? Are there missed angles or opportunities? Could you fill those gaps? How does this information inform your own point of view and messaging?

The results of your analysis may be useful as noted in the above goals, and as groundwork for Technology Facet activities around roadmap development.

Activity adapted from: Alameda, Leeann. "The Power of Competitor Analysis for Nonprofits." NonProfit PRO. Accessed February 18, 2021. https://www.nonprofitpro.com/article/the-power-of-competitor-analysis-for-nonprofits-and-how-to-get-started/.



Activity 5: Buy a Feature Game

Goals

1. Help users prioritize among a list of pre-defined (but not yet developed) features
2. Promote cooperation among end users

Prerequisites

Example: None

Who Should Participate?

Program management (tactical thinkers), Program staff (operational experience), End users

Length

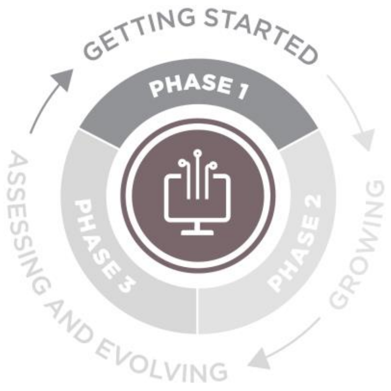
60 minutes

Activity Overview

This is a game to get end users to prioritize among a list of pre-defined (but not yet existing) features. Users get a limited budget of play money to fund their preferred features (a sample game board is linked in the instructions below). The instructions are divided into two sections, the first that a subset of program management representatives should complete, and the second with a group of end users.

Program Team Activity Instructions - Complete First

1. Program management representatives (e.g. PM, Tech Lead) work together to create a list of feature requests or potential roadmap additions for end users to help prioritize.
2. Define a budget for your participants, e.g. $100 per person or organization.
3. If holding the activity virtually, create a spreadsheet to track the contributions. Suggested setup:
   a. In the first worksheet, create a list of all features you'd like participants to spend money on.
   b. In subsequent worksheets, copy that list of features, and add a column for participants to note how much of their budget they would spend. It's helpful to add a sum at the bottom so they can see how much they've spent.
   c. Back in the first worksheet, add a column that rolls up the dollars spent by participants in each individual worksheet to see which features were most highly prioritized.
   d. A sample worksheet with pre-set calculations is available here.
4. If holding the activity in person, write the names of the features on cards, and put a small container in front of each card. Provide participants with a stack of play money and have them contribute to each feature.
5. Schedule the activity and invite guests.



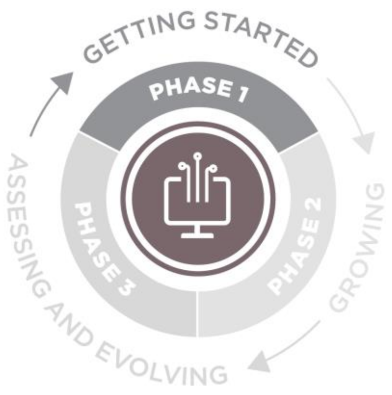
Activity 5: Buy a Feature Game

End User Instructions

1. **Have the participants buy features**. Invite participants to allocate their budget among the features they want.
    a. Optional: Allow users to suggest new features that are not on the list.
    b. Note: This can be done synchronously as a group or asynchronously in which your invitees are given a set time (e.g.) a week in which to purchase their features.
2. **Analyze outcomes**. Analyze which features get the most attention, which are funded, and which ones are not.

Output

A prioritized list of not-yet-developed features to assist with future roadmap development.



TECHNOLOGY

Phase I: Laying the Groundwork

Activity 6: Community QA Testing

Goals

1. Review program's current QA practices and evaluate whether they are structured in a manner that can be supported by the community
2. Identify places for improvement
3. Conduct a successful round of Community QA

Prerequisites

None, but Activity: Recognition and Contributions may be useful.

Who Should Participate?

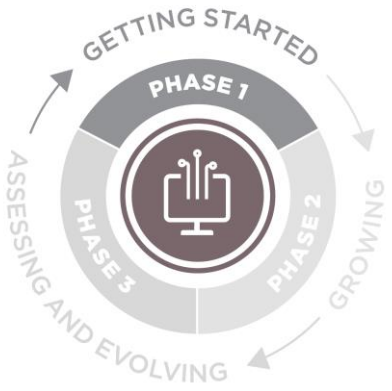
Program management (tactical thinkers)

Length

X minutes

Activity Instructions

1. Fill out the checklist on pages 2-3. Rate whether your program has these elements in place, does not, or they're in progress. You can also note if you are unsure, or if the metric does not apply.
2. Work together to identify 3-5 elements in the checklist that could be improved before your next release. Plotting elements on an impact/effort matrix (example on page 4) can aid prioritization.
3. Identify who on the program team will be accountable for the element - either taking on the responsibility for creating/improving the element or assigning the task to someone else and following up on its completion.

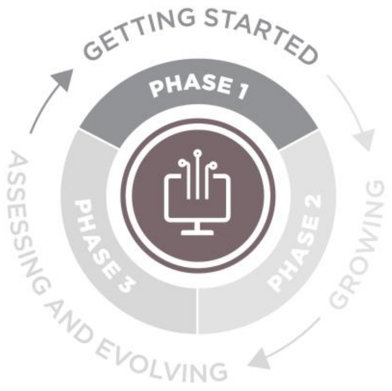


TECHNOLOGY

Phase I: Laying the Groundwork

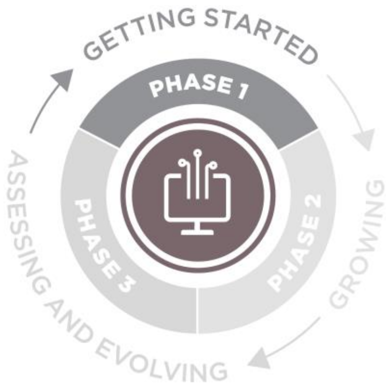
Activity 6: Community QA Testing

Testing tools	Yes	No	In progress	Unsure	N/A
Clear locations for QA communication / documentation: QA docs for program staff QA tests for end users Bug tracker instructions Bug tracker Q&A / Help Other ________					
Instructions for which platform to use for which purpose *Jira for bugs, Slack for questions, wiki for test plans, etc.*					
Volunteers	Yes	No	In progress	Unsure	N/A
Cadre of dedicated testers *Maintain, review list on regular basis*					
Orientation for new testers					
Clear instructions for new and returning testers *How to choose a test, file a bug, etc.*					
“Good first test” flag for new testers					
Method for ID’ing community members who could take on additional responsibility					
Recruitment plan for new testing volunteers					
Events	Yes	No	In progress	Unsure	N/A
Schedule for QA testing process					
Themed and time-bound QA events *New releases, beta features, usability, etc.*					
Events adapted for different times zones and/or languages					
Event promotion across multiple platforms					
Specific and clearly communicated goals for the event					
Reward / recognition plan					

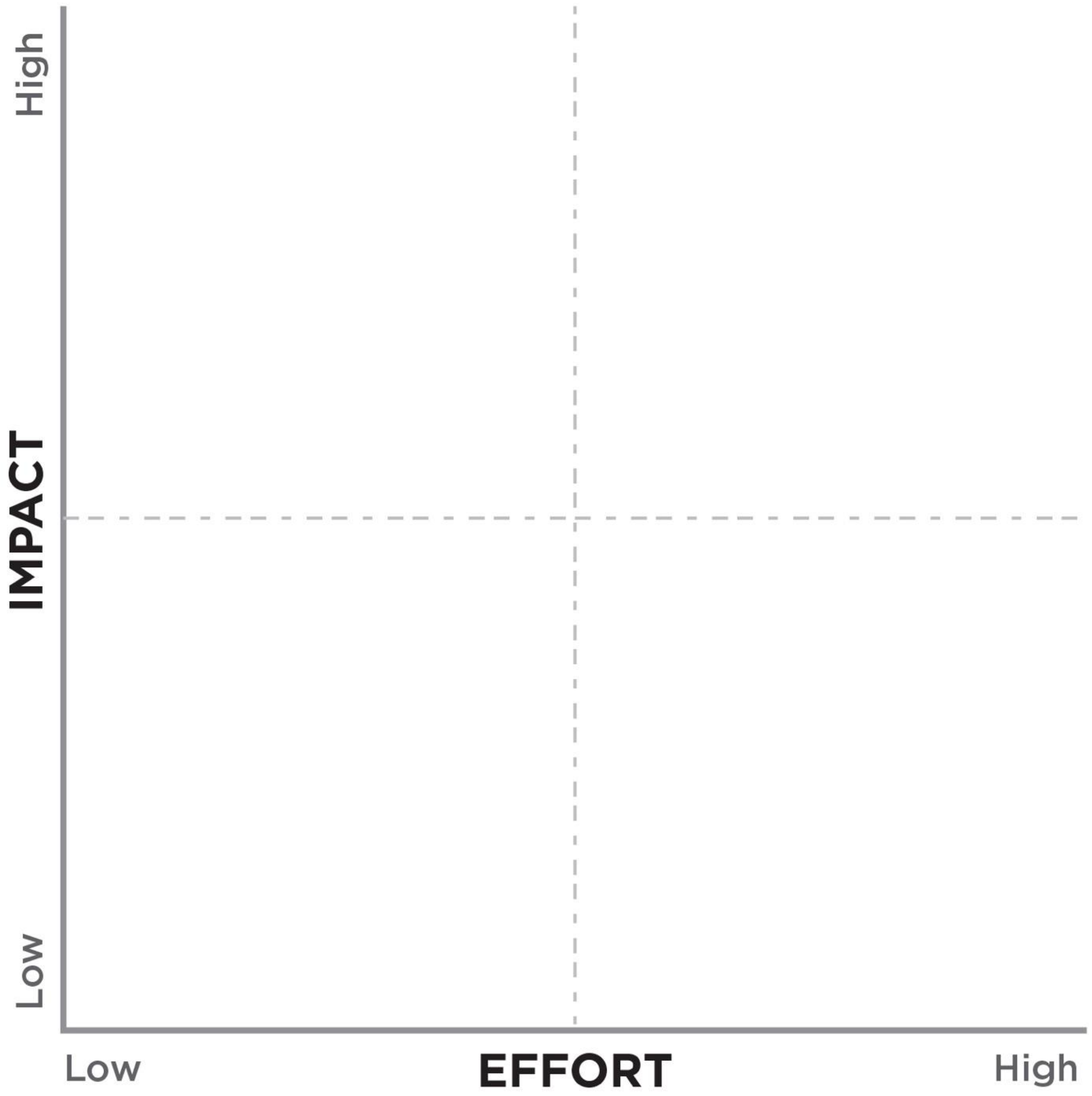


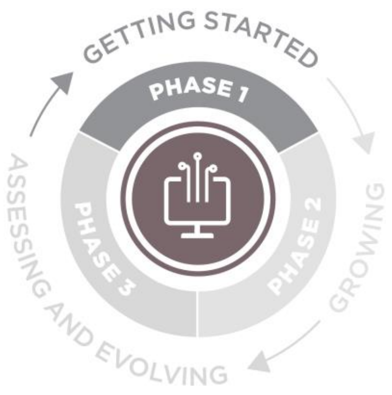
Activity 6: Community QA Testing

Documentation	Yes	No	In progress	Unsure	N/A
Roles and responsibilities *Who manages the events – staff, community members? What are their roles?*					
What is QA / QA testing *Non-jargon essential*					
How to create test plans					
How to QA test plans (really) *Are the steps outlined in the test plan really those taken by the end users?*					
How to organize / hold a testing event					



Activity 6: Community QA Testing





TECHNOLOGY

Phase I: Laying the Groundwork

Activity 7: Docs Friction Logging

Goals

1. Understand how usable your current documentation is via friction logging with documentation users.

Prerequisites

None

Who Should Participate?

Documentation creators and consumers

Length

30-60 minutes for documentation consumer, additional time for creators to review results and update documentation as needed.

Definition

A friction log is a document that lists the elements that make a tool hard to use but frames it around a narrative or customer use case. The critical difference between a friction log and a bug list is that the friction log puts the issues in context. It tells the story of the entire user experience from start to finish.

Pre-Work

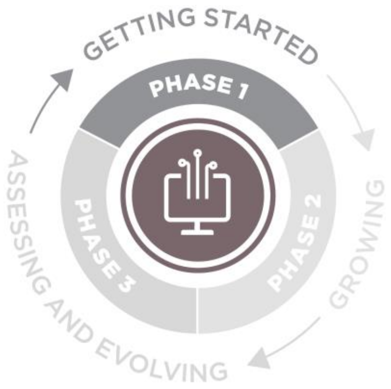
Decide which elements of your current end-user documentation you'd like to evaluate. You may wish to evaluate "user manual" type documentation for your application (e.g. how to create an account or how to upload an image), or program-focused documentation (e.g. how to contribute code or how to submit a bug report). The most useful friction logs will evaluate documentation that lots of folks are likely to use.

A good example of a friction log can be found about halfway down this blog post:

https://www.trychameleon.com/blog/friction-logs

Instructions for Program Staff

1. Determine which elements of your current end-user documentation you'd like to evaluate.
2. Identify community members to serve as loggers. Make choices based on what you're testing; and try not to choose people who are so familiar with the system that they'd know how to skip over the hard or confusing parts.
3. Create a friction log template (example on page 4 below). Include at least the following questions:
   a. Logger's name
   b. Platform/language/browser (if relevant)
   c. Date
   d. Product used for the log (if your program supports >1)

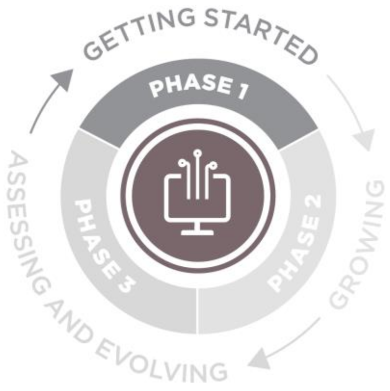


   e. Brief description - no more than two short sentences - of the scenario/use case the logger is following the docs for - such as:
      i. I want to figure out how to contribute code to the platform.
      ii. I am trying to add a new user and give them account permissions.

4. Include a link to the documentation (or attachment if it's not available online)
5. Provide the friction logger with the logging template
6. Once the logger has completed the work, evaluate the comments, and determine if any changes or improvements need to be made to the documentation

Instructions for Friction Logger

1. Using the provided template and following the instructions in the documentation, create a log of what you did to complete your scenario / use case, and any reactions you had while doing so. For example:
   a. If you searched a wiki or GitHub repository, write down the search terms
   b. If you typed commands into a command line, copy and paste your command into the log
   c. Make note of any places where the documentation was not clear, or the instructions didn't work as written
2. Record your reactions for each step, such as "Now I'm frustrated," or "Copied and pasted this from the docs, didn't bother reading the prose."
3. Stoplight-colored highlights (red, yellow, green) in the log can be used to point out particularly excellent or problematic parts of the experience.
   a. Green: Something delightful. Maybe there was a great example, or something worked right out of the box without requiring extra configuration.
   b. Yellow: Something that was frustrating. Perhaps you couldn't find relevant docs after 10 minutes of searching.
   c. Red: Places where you would have given up if this wasn't your job.



TECHNOLOGY

Phase I: Laying the Groundwork

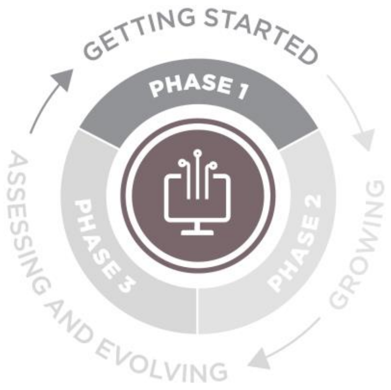
Activity 7: Docs Friction Logging

Example

Scenario: Create a Microsoft Teams account

Steps:

1. Use Google to look up “Microsoft Teams”
2. Click on the first result - “Video Conferencing, Meetings, Calling | Microsoft Teams”
3. “Sign up for free” is the first thing I notice on the screen, so I click on that button
4. I enter my email and click Next
5. I am using this to participate in volunteering, so I select “For work and organizations”
6. Enter password? They want the password to my email account, or do they want me to create password for this Teams account? This is confusing.



TECHNOLOGY

Phase I: Laying the Groundwork

Activity 7: Docs Friction Logging

Friction Log Template

Name:

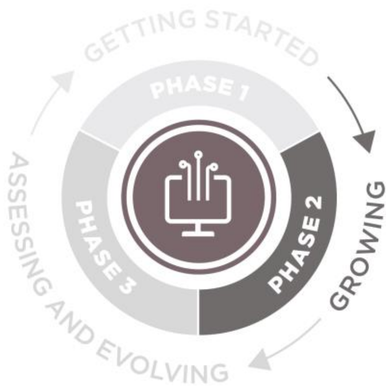
Platform/Language/Browser:

Date:

Platform:

Scenario / Use Case (two short sentences):

Create a narrative that tells the entire story, from start to finish, of your scenario or user story. Include every step you took and reaction you had. Stoplight-colored highlights (red, yellow, green) or prefixes in the log can be used to point out particularly excellent or problematic parts of the experience: ***Green:*** *Something delightful,* ***Yellow:*** *Something frustrating,* ***Red:*** *Places where you would have given up if this wasn't your job.*



TECHNOLOGY

Phase II: Expanding and Integrating

Activity 8: Shadow Observations

Goals

1. Gain insight about users' jobs, pains, and gains
2. Learn about users' day-to-day work in a real-life setting
3. Incorporate end-users experience into program planning

Prerequisites

None

Who Should Participate?

Program management (tactical thinkers)

Length

60 minutes per shadow

Activity Instructions

In shadow observations, one or more members of the program staff observe an end user interacting with the software. Generally, the people observing the user don't interfere with the user's actions - the idea is to see how they go about their daily work as normal.

This activity can be done in-person, or online via screen-sharing.

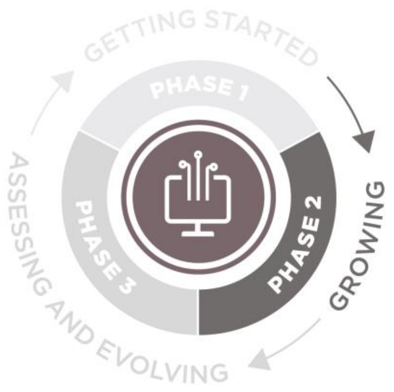
Pre-Work

Work with program colleagues to:

- Determine the goal of the shadowing exercise - do you want to see how a new feature is being used, get information to help with prioritization, etc.?
- Invite the community to volunteer for shadowing. Your invitation should include:
  - Brief description of what shadow observation is
  - Length of observation
  - Platform
  - Whether it will be recorded (recommended if online)
  - What specific interactions you'd like to see, e.g. logging in, adding content, running a report, etc.

During the Observation

- Beyond the initial prompt (show me how you…), try not to interfere with the user, just let them user the software normally
- Ask the user to narrate their actions as they complete them
- Keep track of your observations as they happen, the template below may be helpful
- Hold questions until the end when the shadowing is complete



TECHNOLOGY

Phase II: Expanding and Integrating

Activity 8: Shadow Observations

After the Observation (Still with User)

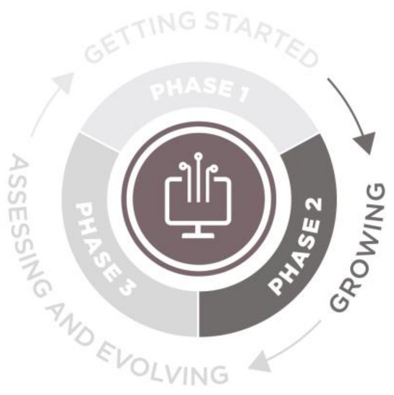
- Ask questions! Ask the user to clarify anything you didn't understand or would like more detail about.
  - Consider using a technique such as the Five Whys, which asks "why" questions in response to five consecutive answers. This prompts the user to examine and express the underlying reasons for their actions.

After the Observation (No Longer with User)

- What did you learn?
- What positive interactions did you observe?
- What pain points did you observe?
- How can you fold your findings into future road mapping and prioritization exercises?
- Summarize the high-level results and share them with the community
- Work to prioritize concerns/needs and fold into road mapping activities

Sample Notes Template

Time	Activity (What I See)	Notes (What I Think)



TECHNOLOGY

Phase II: Expanding and Integrating

Activity 9: Recognition & Contributions

Goals

1. Understand how the program currently recognizes contributions
2. Understand the kind of recognition that is most appreciated/motivating to each kind of contributor
3. Discover how the program can improve in this area

Prerequisites

Tech Activity: Who are Your Technology Stakeholders?

Who Should Participate?

Program management (tactical thinkers), Program staff (operational experience)

Length

60-120 minutes

Activity Instructions

As a group:

1. Discuss the 6Rs (listed below).
2. Which of the elements are currently incorporated in your program?
3. Which are not currently incorporated?

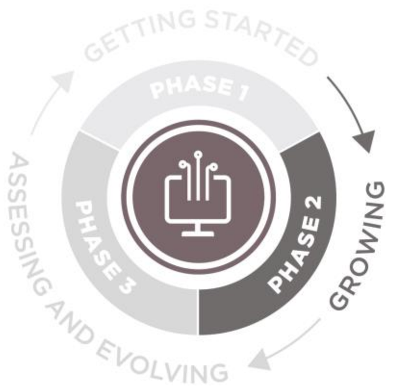
Divide into small groups of 4-5 people. In each group:

1. Select a specific stakeholder group from Tech Activity 1 (linked above - it's ok for the same stakeholder groups to be discussed by more than one small group)
2. Take 15 minutes to talk through what types of recognition might be important for the chosen stakeholder group
3. Select a representative to report back to the larger group

As a group:

1. Hear reports back from small groups
2. Discuss themes and gaps that need to be addressed to keep current contributors engaged and to entice new contributors
3. Prioritize and create an action plan to incorporate new forms of recognition
4. Who will be accountable for ensuring the plan is created / carried out?
5. Who will be responsible for doing the work?

For the future / Potential next steps: The information gathered during this exercise can inform working groups or specific outreach activities. Consider if there need to be higher level program changes to accommodate the recognition needs (i.e. more representation on governance groups or different voting rights).



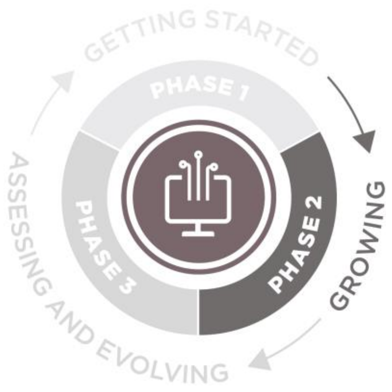
Activity 9: Recognition & Contributions

The 6Rs

Contributions are critical for community and open-source software. Understanding what motivates potential and current contributors is essential.

There are 6 crucial recognition qualities. These are not "one size fits all" but these can be a helpful way to consider what is important to different contributors. They include:

1. **Recognition**: People want to be recognized for their contributions.
2. **Respect**: People want their values, culture, ideas, and time to be respected and considered in the organization's activities.
3. **Role**: People want a clearly meaningful role in the coalition that makes them feel valuable and in which they can make a contribution.
4. **Relationships**: People want the opportunity to establish and build networks both professionally and personally for greater influence and enjoyment.
5. **Reward**: People expect the rewards of participating in a collaborative partnership to outweigh the costs and to benefit from the relationships established.
6. **Results**: People respond to visible results that are clearly linked to outcomes that are important to them and that they can clearly link to their participation in the coalition.



TECHNOLOGY

Phase II: Expanding and Integrating

Activity 10: Value Propositions for Job Descriptions

Goals

1. Create a value proposition for a new program position (e.g. community manager, technical lead) to help articulate significance of allocating resources to program leadership
2. Optional: Create a job description for a new program position

Prerequisites

Job description (optional). If you don't have a job description yet, having samples on hand of job descriptions of similar roles at other projects can be helpful as a starting point.

Who Should Participate?

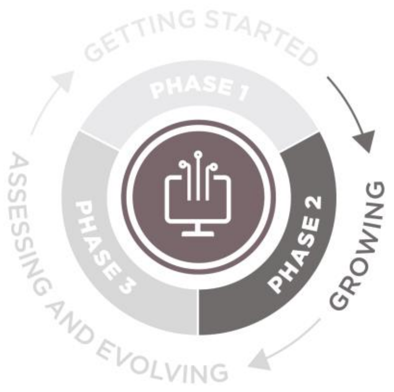
Program management (tactical experience), Program staff (operational experience)

Length

60-90 minutes

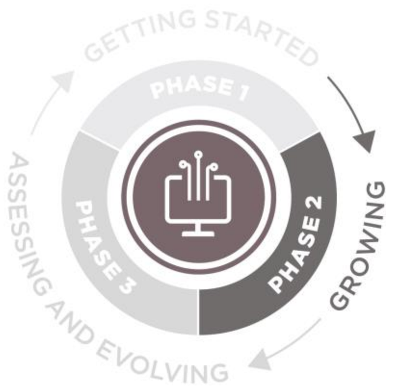
Activity Instructions

1. List the potential responsibilities (i.e. tasks and duties) for the position.
   a. If you already have a job description, you can take the responsibilities list from that.
   b. If you do not already have a job description, this is the place to go broad - not all these elements will make it into a final job description or value proposition.
      i. It can be helpful to set objectives for the number of ideas to be listed and the time to be spent, e.g. "Let's spend 5 minutes coming up with a list of 20 responsibilities / tasks / duties we think this job would cover."
2. Take a quick sticker vote to select the responsibilities that are the most critical / highest priority.
   a. In a sticker vote, each participant is assigned a number of stickers - these can be physical stickers in an in-person event or a specified piece of text (e.g. +1) in a virtual environment. Participants place their stickers or text alongside the options they're voting for, according to the parameters of the exercise (e.g. most important, most likely, most interesting, etc.).
3. For each of your top vote-getters, discuss and capture how the responsibility / task / duty would solve problems faced by your application, program, and/or community or add new benefits. For example, could the person in this role:
   a. Save time and/or resources? Help the project increase resources?
   b. Improve community buy-in and engagement?
   c. Improve the application's quality or functionality?
   d. Eliminate risks the application, program, or community might face?
   e. Help end users use the application more effectively?
   f. Eliminate barriers to adoption?
   g. Create positive social consequences?



Activity 10: Value Propositions for Job Descriptions

4. Rank each of the elements articulated in Item 3 as “essential” or “nice to have.”
5. Finally, collate the “essential” value proposition elements into a value proposition document.
6. If you don’t already have a job description, you can use the top priority elements from Item 2 as the basis for creating one.
7. Share both documents with program leadership.



TECHNOLOGY

Phase II: Expanding and Integrating

Activity 11: Building Welcoming Communities

Goals

1. Identify what documentation your program has that supports growing a successful community
2. Prioritize missing documentation and develop a plan for its development

Prerequisites

Example: None

Who Should Participate?

Program management (tactical thinkers), Program staff (operational experience)

Length

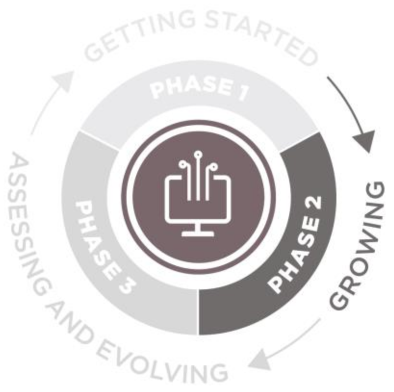
60-90 minutes

Pre-Work

Have a subgroup (2-3 program representatives) complete the checklist on page 2 and bring it to a larger group for broader discussion and prioritization. It is okay to determine that some components are not relevant or propose adjustments.

Activity Instructions

1. As a group, review the submitted checklist. Does everyone agree or not? Are there elements not included in this assessment?
2. Once general consensus on rankings is achieved, lead the group in identifying what components the group considers necessary to work on, then prioritize (as you cannot do everything at once). Plotting elements on an impact/effort matrix (example on page 4) can aid prioritization.
3. Identify who on the program team will be accountable for working on the component - either taking on the responsibility for creating or improving it, or assigning the task to someone else and following up on its completion



Activity 11: Building Welcoming Communities

Building Welcoming Communities Checklist

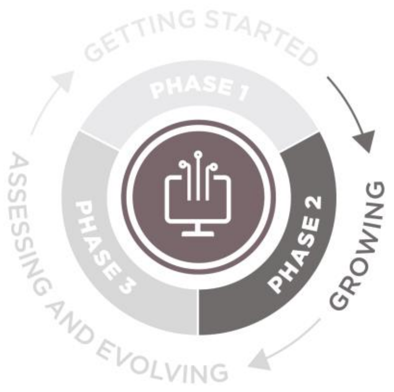
Does your program have the following documentation?

	Yes	No	In Progress	Unsure	N/A
A friendly README					
Clear code examples					
Code contribution guidelines					
Good first issue tags					
Response plan for new contributors					

Growing Your Community

How can you provide your community with a good foundation for community growth? Do you have the following?

	Yes	No	In Progress	Unsure	N/A
Code of Conduct					
Contributors or Authors file					
Regular newsletter or other communication thanking contributors					
Organizational repository (vs. personal)					
Structured roles for newcomers to fill?					
Clear and transparent communication channels?					
Regular schedule and structure for trainings, onboardings, workshop, etc.?					

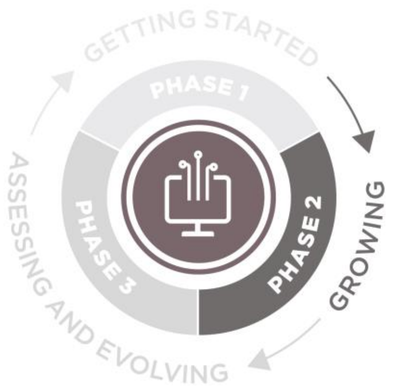


Activity 11: Building Welcoming Communities

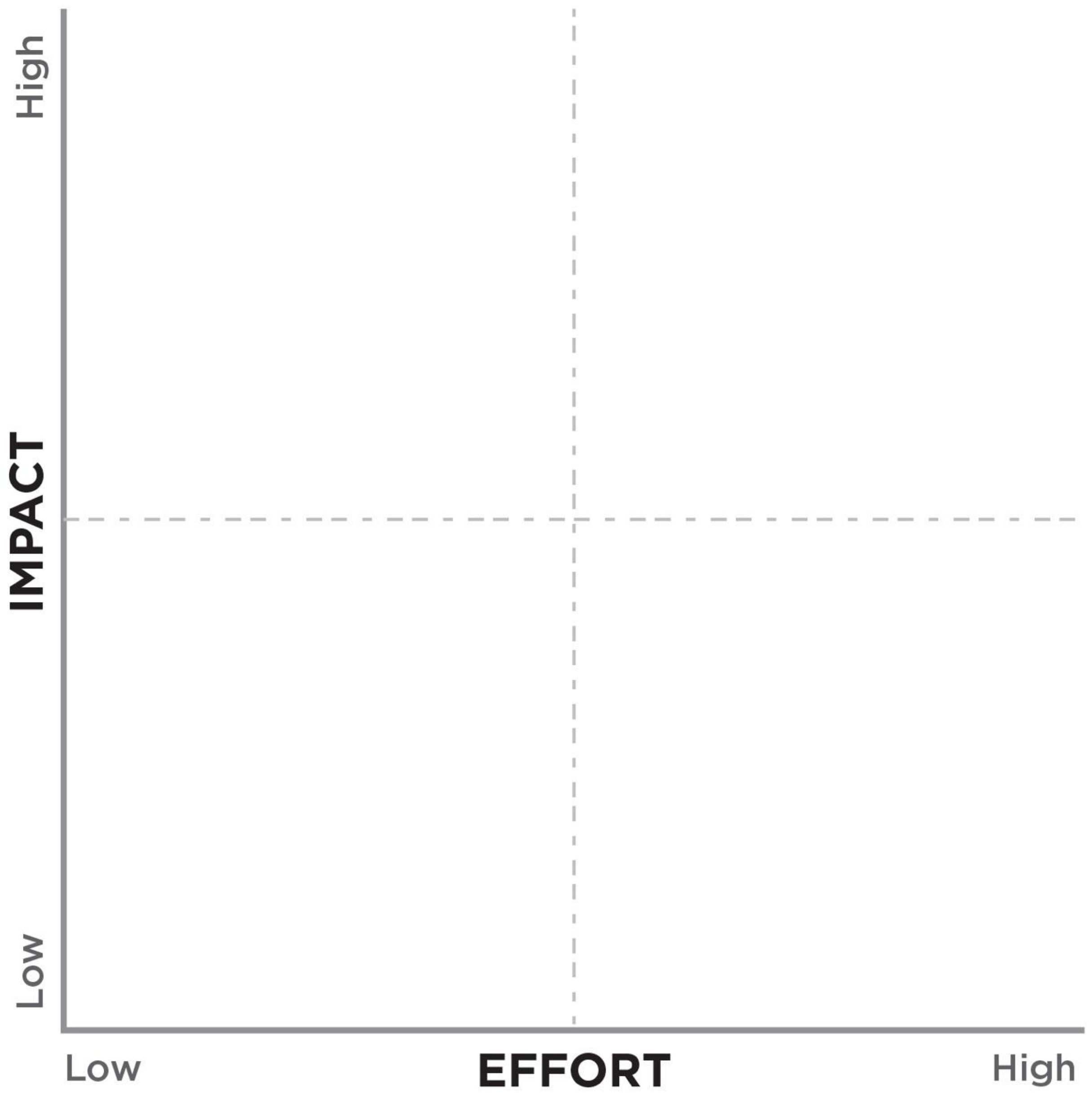
Resolving Conflicts

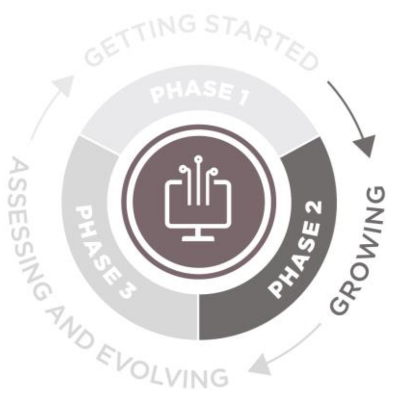
Do you have a plan for conflict resolution as your community grows? Do you have the following helpful tools?

	Yes	No	In Progress	Unsure	N/A
Product vision and roadmap					
Documented decision-making process (e.g. consensus, voting)					
Identified community tiebreaker					



Activity 11: Building Welcoming Communities





TECHNOLOGY

Phase II: Expanding and Integrating

Activity 12: Not Invented Here: Assessing Integration Options

Goals

1. Identify functional requirements or feature requests that may be solved by integration
2. Identify and assess potential integration solutions to satisfy functional requirements or feature requests

Prerequisites

Development Roadmap, Tech Activity: Landscape Analysis

Who Should Participate?

Program leadership (strategic thinkers), Program management (tactical thinkers). This activity is best kicked off with a small group together (in-person or virtually), and then moved to asynchronous work.

Length

Part 1, 60-90 minutes with a small group. Part 2, depends on the scope of the project.

Activity Instructions

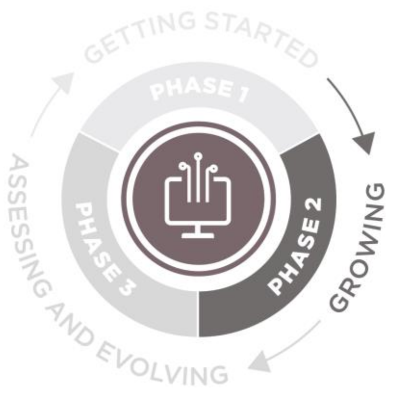
Part 1: Small group together

Identify functional requirements or feature requests on your roadmap that may be solved by integration. Sample questions to ask about proposed features are below; if the answer is yes to a majority of the questions, then integration may be a good idea.

Criteria	Yes	No
Is the request for a brand-new feature?		
Does the functional need have a timeline or expiration date?		
Is the functionality needed quickly, e.g. within one or two quarters?		
…		

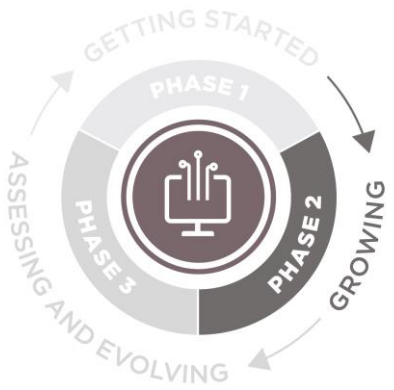
Part 2: Small group asynchronously

- Compare functional requirements with the features/functionality of the solutions identified in your landscape analysis
  - It may be useful to break the functional requirements into user stories to compare
- Assess suitability of potential solutions
  - Does the platform have a compatible license?
  - Is the identified solution designed as a component or amenable to re-use?
- Complete a gap analysis
  - Determine what is missing from your functional requirements
  - Prioritize the gaps with your community - are they must haves?



Activity 12: Not Invented Here: Assessing Integration Options

- Map the integration
    - Create a workflow document that shows how the two systems would interact.
    - Is one system fully contained by the other, or do they just communicate? If the latter, does data flow both ways? Which is the system of record?
- Learn about and communicate with the other program
    - Who is their core community? Does it overlap with yours?
    - Who are the executive, strategic, and tactical leaders?
    - Are they open to an integration? Would their community benefit?
    - Do they have a sustainability plan?
- Determine deal breakers
    - What would stop an integration from being effective?
- Assess outcomes
    - No solution compatible - program will have to build
    - Multiple solutions compatible - assess them all to see which one is the most compatible
    - One best solution - work with governance and technology leadership to determine formal process for moving forward with integration (e.g. is an MOU or other agreement needed?)



TECHNOLOGY

Phase II: Expanding and Integrating

Activity 13: How We Retire Features

Goals

1. Identify elements of the platform that can be retired
2. Develop a plan for communicating retirement plans with the community
3. Communicate the retirement decision to the community

Prerequisites

Example: None

Who Should Participate?

Program management (tactical thinkers), Program staff (operational experience). This activity is best kicked off with a small group together (in-person or virtually), and then moved to asynchronous work.

Length

X minutes

Activity Instructions

Part 1: Small group together

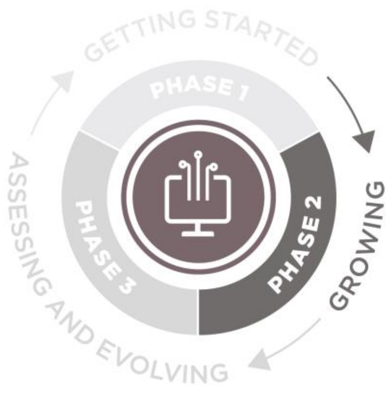
Note: If a specific feature is already under consideration for retirement, you may move directly to Part 2 of the activity.

1. Discuss/brainstorm together a list of features/functionality that may be suitable for sunsetting. Plotting features on an effort/impact matrix (sample on page 3) may help identify elements that are difficult to maintain but do not provide a huge value to users.
2. If there are several, select 1-2 features for further assessment.
3. Determine a timeline for the remainder of the activity, who will be responsible for its completion, and who will be accountable.

Part 2: Small group asynchronously

1. Gather and review data on each of the selected features:
   a. User data
      i. How many users will be affected?
      ii. What percentage of overall users use the feature?
      iii. Did they try it once or did they use it all the time?
      iv. What percentage of users *do not* use the feature?
      v. Why did those users not use it?

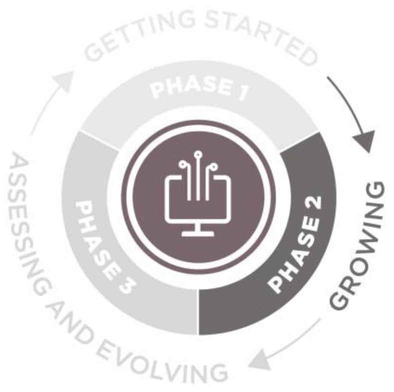
   b. Data on alternative workflows
      i. What specific workflows is the feature supporting? When asking users about how they're using specific features, beware the XY problem; or asking about a user's solution rather than the problem they're trying to solve.
      ii. What alternatives are there to complete the workflow?



   c. Data on organizational effort
      i. What resources (engineering, support) does the feature require?
      ii. What percentage of product use and/or revenue comes from the feature?
      iii. What is the expected cost of keeping it vs. the potential loss of removing it?
      iv. Is the product adding or distracting from our program mission?
2. Make a recommendation to retire/not retire features based on the data and information gathered.

If, after the above elements are complete, the decision is made to sunset a feature, move on to:

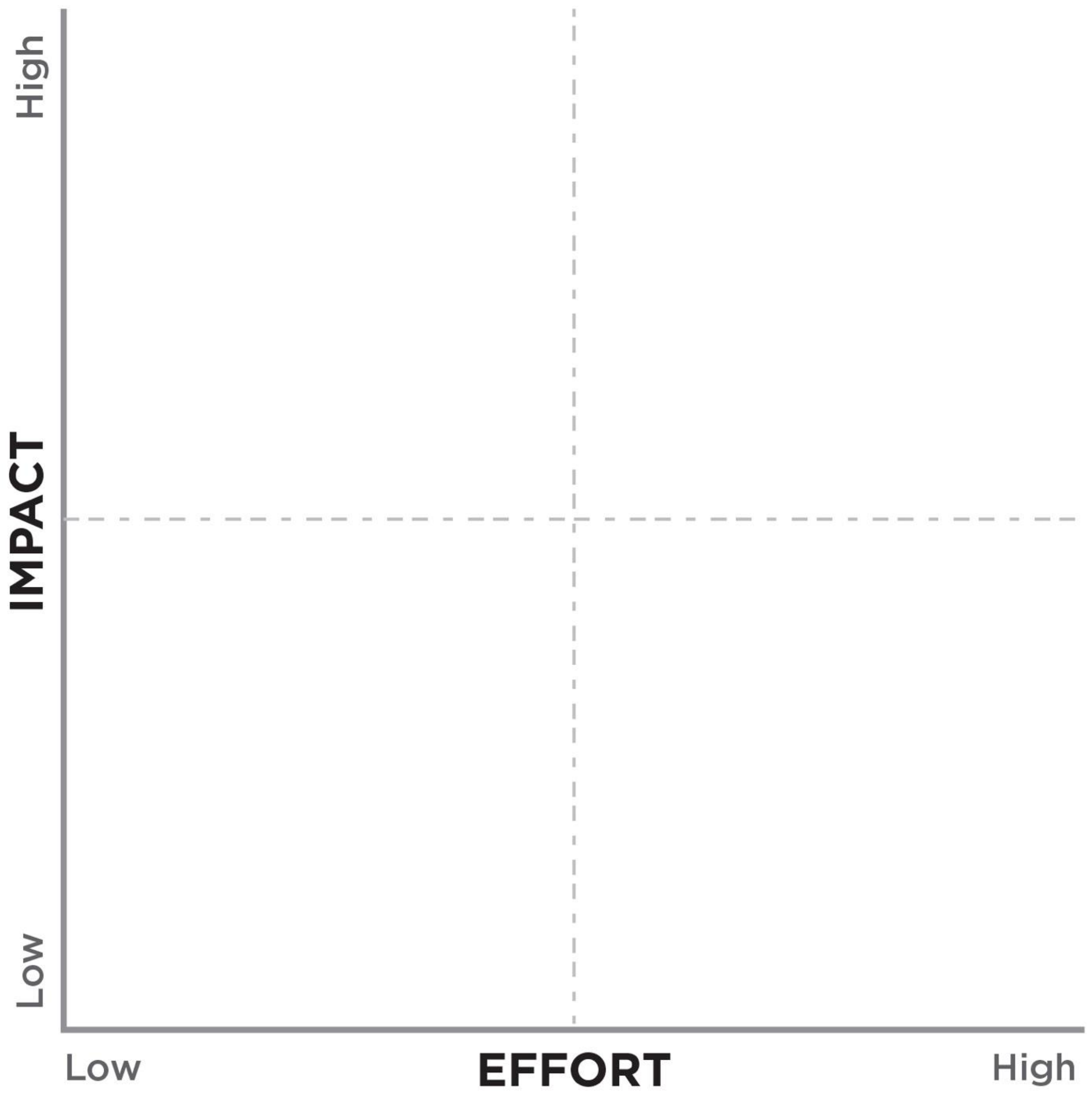
3. Gain approvals from the appropriate governance/leadership representatives.
4. Develop a sunset communication plan
   a. See Technology Activity: Developing an End-of-Life Communications Plan
5. Communicate
   a. Internal
      i. Ensure all program team members are aware of the change and the key details - why, when, alternatives
      ii. Update support documentation to clarify that a feature is no longer available, and what alternative workflows are available.
   b. External
      i. Reach out to affected users with a simple, concise message - enough detail so that nothing is unclear, dates and times the feature will be discontinued, and recommendations for alternative workflows.
      ii. Reach out via different channels: emails, in-application messaging if available, phone calls, social media.
      iii. Monitor feedback in the days/weeks following the announcement.

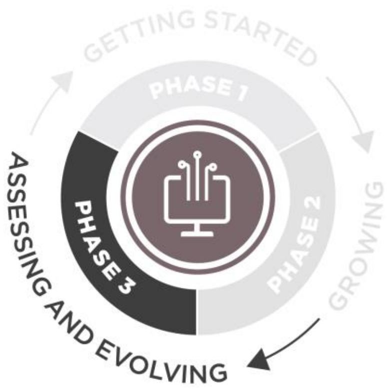


TECHNOLOGY

Phase II: Expanding and Integrating

Activity 13: How We Retire Features





TECHNOLOGY

Phase III: Preparing for Change

Activity 14: Technology Stakeholders: Does the Shoe Still Fit?

Goals

1. Re-identify and re-prioritize the community's technology stakeholders
2. Create a high-level map of the technology stakeholder's characteristics
3. Support program team's work developing strategies to advance technology sustainability

Prerequisites

Program Mission and Vision, Results of Tech Activity: Who are Your Technology Stakeholders? (if available)

Who Should Participate?

Program leadership (strategic thinkers), Program management (tactical thinkers), Program staff (operational experience)

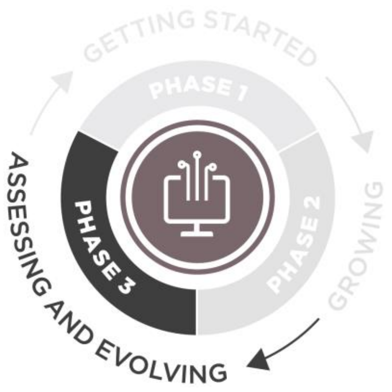
Length

90-120 minutes

Activity Instructions

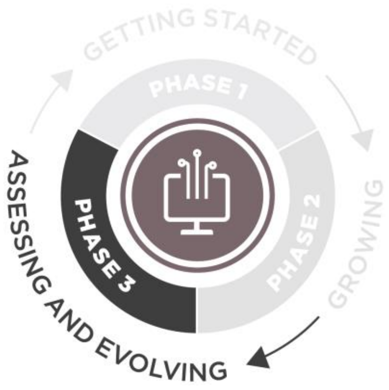
NB: If you have recently (<18 months) completed the activity Who Are Your Technology Stakeholders, gather the results and skip to Step 4.

1. On a whiteboard or shared online document, identify your potential technology stakeholder groups. Using sticky notes (real or virtual) works best, as ideas will be moved around in later steps.
   - It's ok to be granular - your team will eliminate duplicates / combine stakeholders in the next step.
   - Stakeholder examples: end users, QA testers, code contributors, software engineers, sysadmins/IT staff at institutions using the software, program staff at related OSS programs, functional requirements contributors, potential home organizations
2. Move the physical/virtual sticky notes around to categorize your stakeholders into groups.
   - Group examples: Technical contributors, Non-technical contributors, Service providers, etc.
3. Select 3-5 groups to prioritize over the next year. An interest/influence grid or sticker vote may be helpful in prioritizing.
   - An interest/influence grid plots stakeholder groups against two axes: Interest/Availability and Influence and then suggests a level of engagement based on their place on the grid. A sample grid is on page 3.
   - In a sticker vote, each participant is assigned a number of stickers - these can be physical stickers in an in-person event or a specified piece of text (e.g. +1) in a virtual environment. Participants place their stickers or text alongside the options they're voting for, according to the parameters of the exercise (e.g. most important, most likely, most interesting, etc.).

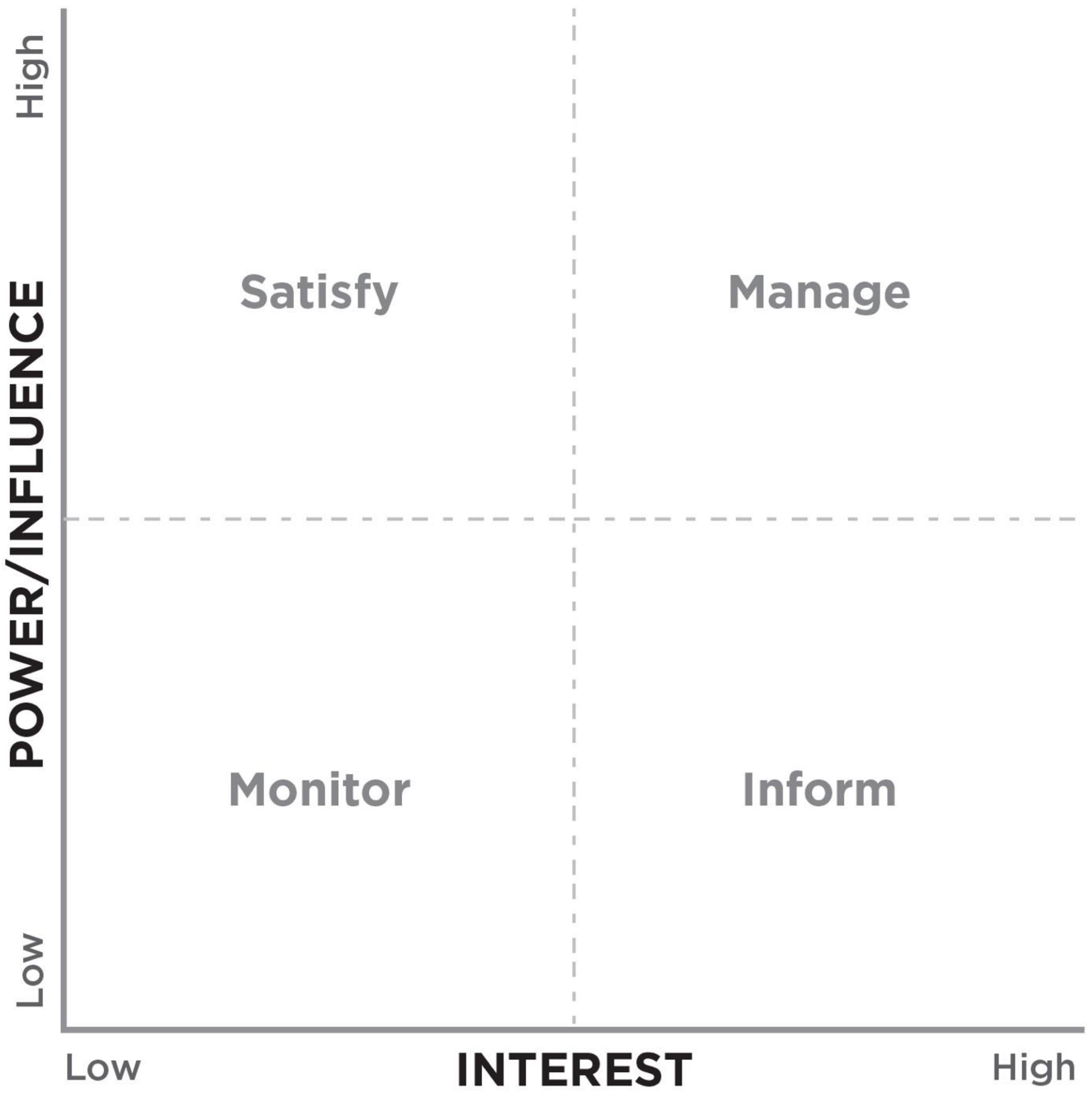


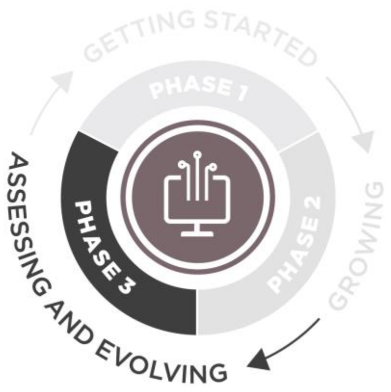
Activity 14: Technology Stakeholders: Does the Shoe Still Fit?

4. If you have the results from an earlier version of this activity (i.e. undertaken during Phase 1), answer the following questions:
   - What has changed about the stakeholder map? Are there new stakeholders or have some dropped off?
   - How have your program's priorities changed? Have different stakeholders groups become more or less important?
   - If there are changes, what effect might they have on the program's priorities?
5. If time allows, for each prioritized stakeholder group in your diagram, discuss the following questions:
   - What are the goals for each user group or technology stakeholder group? Are we clear on what those are? Have they changed over the last 12-24 months?
   - Are there shared or related goals across stakeholder groups? What are the opportunities and areas of collaboration? How can our community work together to create and achieve things?
   - What technology skills are required to make our mission a reality? Which stakeholder groups have these skills?



Activity 14: Technology Stakeholders: Does the Shoe Still Fit?





TECHNOLOGY

Phase III: Preparing for Change

Activity 15: Catastrophizing: Tech Edition

Goals

1. Understand how the program's current technical staff, stack, and roadmap work with unexpected issues (catastrophes)
2. Help programs identify how resilient their staff, stack, and long-range technical strategy are

Prerequisites

None

Who Should Participate?

Program leadership (strategic thinkers), Program management (tactical), Program staff (operational experience)

Length

60 minutes

Activity Instructions

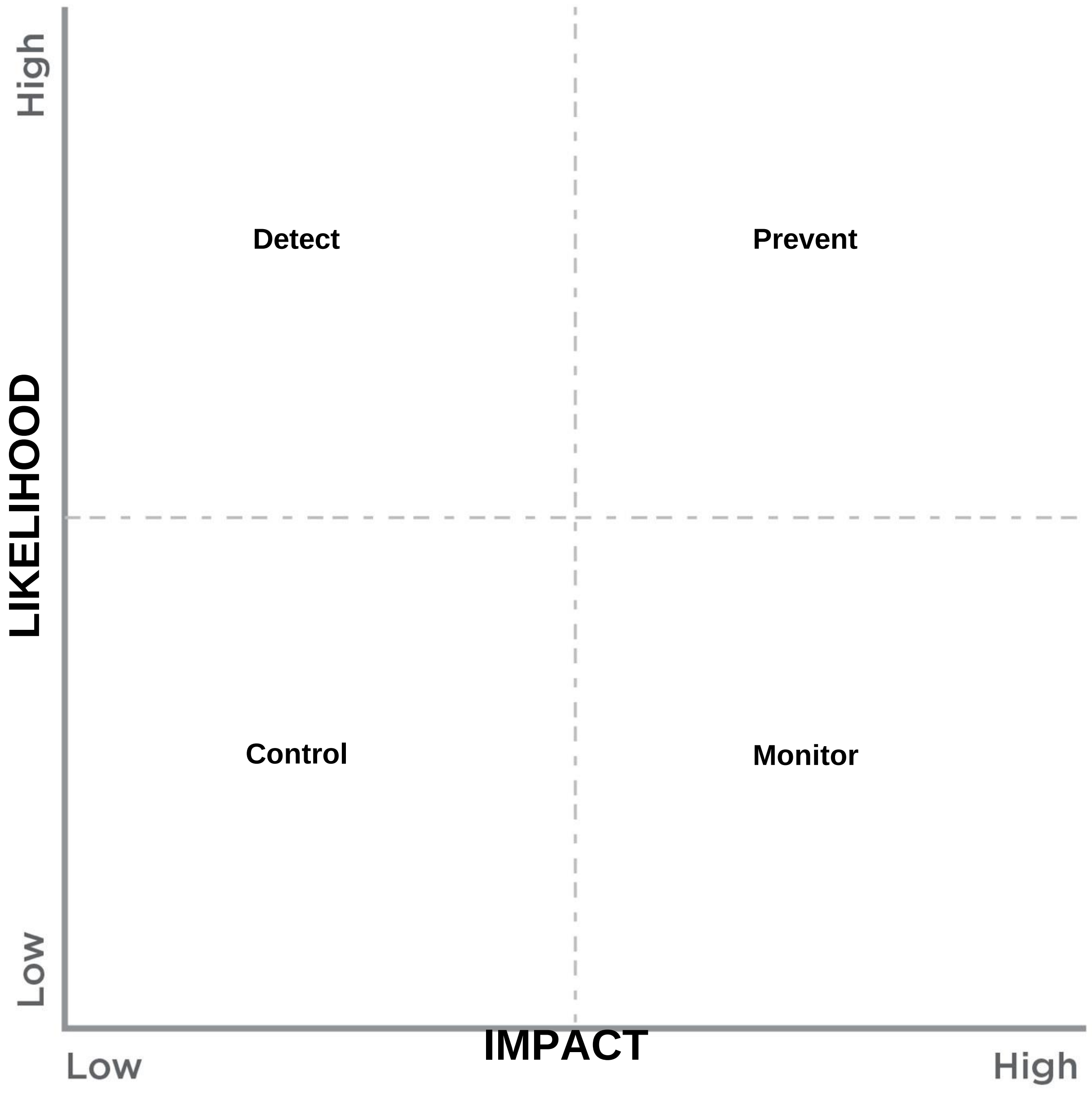
1. Brainstorm a list of technical "catastrophes" that would have a significant impact on your program's ability to fulfill its mission. The examples or suggestions can be outlandish - the idea is to think of catastrophic events and how they would be handled.
   a. Defer judgment - make it clear that stakeholders can say whatever they like.
   b. Encourage wild ideas - these can lead to creative leaps!
   c. Build on the ideas of others - encourage stakeholders to "and" each other's ideas.
   d. Set objectives for the number of ideas to be listed and the time to be spent, e.g. "Let's spend 10 minutes coming up with a list of 30 new ideas."

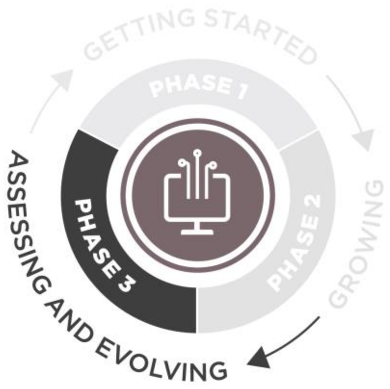
If no immediate suggestions are given, the activity facilitator can seed the discussion with examples such as:

- What if your technical lead won the lottery and moved to Tahiti?
- What if a key element of your technology stack was discontinued?
- What if a service provider forked the code and became a competitor?

2. Once there is a list of several catastrophes, have participants work together to plot the catastrophes on a risk map with axes of Likelihood and Impact (example on page 2). Once complete, move to Tech Activity: Catastrophizing - Tech Edition Part 2.

Activity 15: Catastrophizing: Tech Edition





TECHNOLOGY

Phase III: Preparing for Change

Activity 16: List of Dreams

Goals

1. Articulate what future directions your core application could take
2. Plan for and prioritize professional development opportunities for program stakeholders (staff, contributors, trainers, etc.)

Prerequisites

Agreement from program leadership to allow program staff to devote a specific amount of time to professional development.

Who Should Participate?

Program management (tactical thinkers), Program staff (operational expertise), and other interested Stakeholders (e.g. end users, contributors). This activity is best kicked off with a large group together (in-person or virtually) and then moved to asynchronous work with specific assignments.

Length

90-120 minutes for initial brainstorm and discussion

Activity Instructions

Large group together

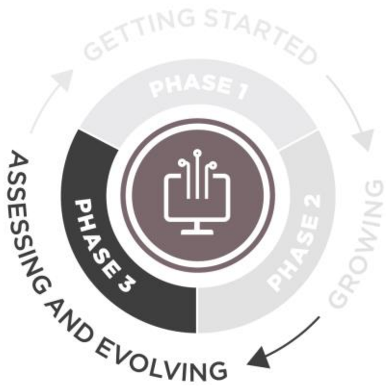
1. Brainstorm ideas for what your program's core application could do, if it could do anything. Ideas should be unconstrained by current features, technology capabilities, funding, etc.
    a. Defer judgment - make it clear that stakeholders can say whatever they like.
    b. Encourage wild ideas - these can lead to creative leaps!
    c. Build on the ideas of others – encourage "and-ing" each other's ideas.
    d. Set objectives for the number of ideas to be listed and the time to be spent, e.g. "Let's spend 10 minutes coming up with a list of 30 new ideas."
2. Take a few minutes to group similar items together.
3. After the brainstorm is complete, have participants sticker vote on the ideas that resonate the most with them - assign each participant a set number of votes to assign.
    a. In a sticker vote, each participant is assigned a number of stickers - these can be physical stickers in an in-person event or a specified piece of text (e.g. +1) in a virtual environment. Participants place their stickers or text alongside the options they're voting for, according to the parameters of the exercise.
4. For the top three or four vote getters, discuss how the program could make it a reality. What new skills or technologies might be necessary?
5. Assign each prioritized skill or technology to a program staff person (as the skill or technology aligns with their interests and capabilities).

Individually, as determined by large group discussion and vote:

As determined by project leadership, devote some amount of time - a week, a sprint, etc. - to investigating that skill or technology.

Each participant in professional development activities:

Report back to program management/leadership. Would adding this new skill or technology help achieve some of the "out there" ideas listed in the brainstorming activity. If not, why not?



TECHNOLOGY

Phase III: Preparing for Change

Activity 17: Catastrophizing: Tech Edition (Part Two)

Goals

1. Turn the issues identified in Part 1 of Catastrophizing into actionable plans.

Prerequisites

Tech Activity: Catastrophizing - Tech Edition Part 1

Who Should Participate?

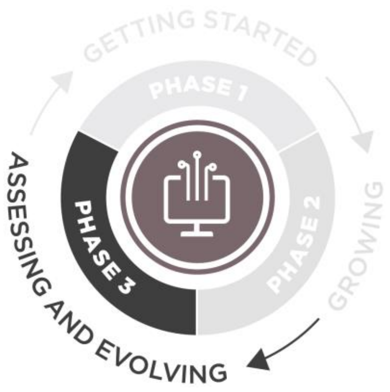
Questions answered by program management, recommendations discussed and/or action on by program leadership.

Length

60 minutes

Activity Instructions

1. Start with the risk matrix developed in Tech Activity: Catastrophizing - Tech Edition Part 1.
2. For at least each high likelihood / high impact catastrophe, talk through the following questions. The template on page two may help organize your thoughts.
    a. What is the catastrophe? What is your definition of the problem?
    b. So what? What are the actual and potential implications?
        i. Think about balance between being prepared and being practical.
    c. Now what? What actions could/should we take going forward?
        i. Are there short-term solutions that can take place while longer-term solutions are being worked on?
        ii. Can we identify stakeholders we can reach out to for help?
3. Once the template is filled in for at least all the elements in the high likelihood / high impact elements, share with program leadership for discussion and/or development of an action plan.



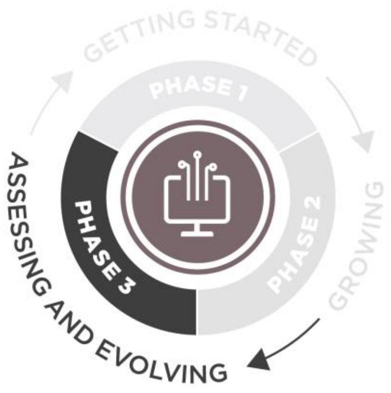
TECHNOLOGY

Phase III: Preparing for Change

Activity 17: Catastrophizing: Tech Edition (Part Two)

Catastrophe Management Template

What is the catastrophe? *What is your definition of the problem?*	**So what?** *What are actual and potential implications?*	**Now what?** *What actions could/should we take going forward?*



TECHNOLOGY

Phase III: Preparing for Change

Activity 18: Technical Skills Inventory (Part Two)

Goals

1. Update your program's inventory of what technical skills are needed for technical staff, contributors, and users to develop, support, and maintain the platform
2. Identify new skills that your program will need to acquire or old skills that can be phased out as the program advances

Prerequisites

Technical Skills Part One, Catastrophizing (Tech) Part One, Catastrophizing (Tech) Part Two.

Who Should Participate?

Program management (tactical thinkers), Program staff (operational experience), Code contributors

Length

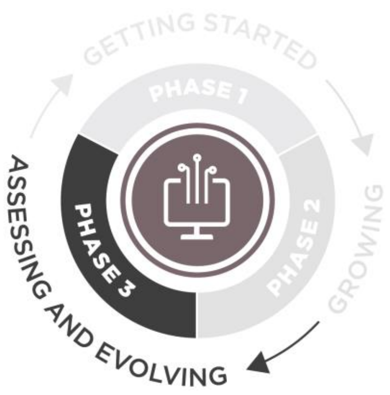
This activity does not need to be done as a group, it can be completed asynchronously/ collaboratively in a shared document.

Activity Instructions

1. Begin with the Technical Inventory created as a result of Technical Skills Inventory: Part One. If your program has not yet completed that Activity, follow the instructions to create a baseline Technical Skills Inventory.
2. Once the Skills Inventory for your existing platform is complete, update the Inventory based on the results of your Catastrophizing exercises. Note in the inventory if the skill is needed now or if one of the catastrophes happens.

Once complete, the inventory may be used for:

- Roadmap planning: ensuring that major deliverables on the roadmap are not all clustered around a certain skillset (and therefore a certain person)
- Roadmap planning: allowing time for professional development to acquire new skills or identify community members with desired skills
- Job descriptions: update job descriptions for existing or potential positions to include the new skills
- Community building: If users self-hosting is important to your program, the skills inventory can be used to compare skills required to install, upgrade, and maintain the software against skills that your end users have or have access to in their organizations



Activity 18: Technical Skills Inventory (Part Two)

Example Skills Inventory

Example roles (depending on tech stack, may need to qualify with frontend, backend, etc.)

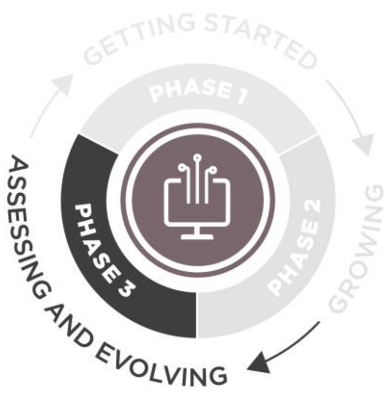
Technical lead, Code contributor, System administrator (e.g. install, upgrade), End user, Migration specialist, Technical support for End user

Example skills (include specific tools in table, e.g. HTML for Web design or development)

Web design, Web development, Assistive technology, Database, Data warehousing, Data analysis, GIS, Platform/OS, Quality assurance/Testing, Reporting, Security, Server

		Skill Level			**Now or Later?**
Role	Required skill	Novice	Intermediate	Advanced	
Developer: Front end	*Angular/Typescript* *HTML* *Bootstrap* *SAAS/CSS*	*X* *X* 	 *X* *X*		*Now*
System administrator (install, upgrade)	*Command line* *Package manager*		*X* *X*		*Catastrophe only*
Report writer	*SQL* *Crystal Reports*			*X* *X*	
…	*…*				

Examples in *blue italics.*



TECHNOLOGY

Phase III: Preparing for Change

Activity 19: Developing an End-of-Life Communications Plan

Goals

1. Develop plan to communicate end-of-life decisions with community

Prerequisites

None, although this guide presumes that your team has already decided that a project, program, or platform will be ending.

Who Should Participate?

Program leadership (strategic thinkers), Program management (tactical thinkers)

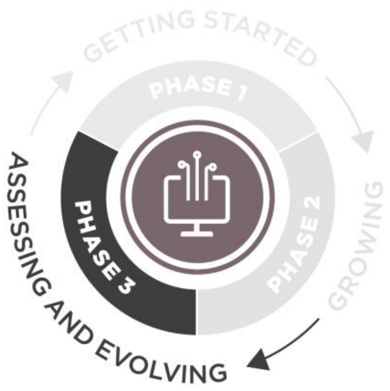
Length

120 minutes

Activity Instructions

As a group, discuss and answer the following questions. The included matrix can help organize your decisions.

1. Who is responsible for implementing the communications plan and ensuring its success?
2. What is being retired? The entire program? An application? Part of an application? The repository, mailing lists, website, or other online forums? Communications should include a wind-down plan and timeline for each element.
   a. Be very clear about what is sunsetting - have answers ready for what you anticipate to be common misperceptions
   b. Be prepared to respond to controversies / negative feedback
3. What do you want to happen as a result of each communication? Let your stakeholders know the purpose of the communication upfront - is it just for information, or do they need to take action?
4. Who is responsible for preparing and delivering communications? Who is on hand to proofread and ensure the message, and any necessary actions on the part of stakeholders, is clear?
5. Who are the stakeholders who need to receive the information? Is the information the same for each stakeholder group?
6. What is your communications timeline? Different communications might be timed to different elements of the wind-down, such as the initial announcement about end of life and then follow-ups about significant events such as the last major release, last maintenance release, end of support, etc.
7. How will you share the information to ensure that all stakeholders receive it? Messages may need to be repeated in multiple formats across multiple channels.
8. Who will be responsible for following up to ensure that messages were received?

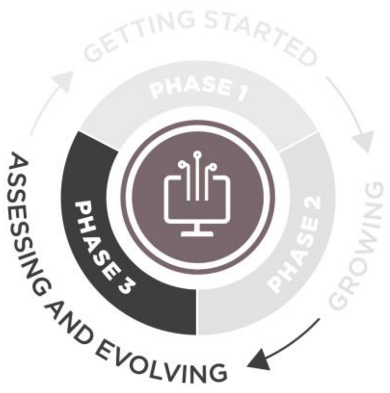


Activity 19: Developing an End-of-Life Communications Plan

Sample Communications Planning Matrix

Description - what type of comm?	Frequency – how often?	Method / Channel	Audience / Recipient	Owner / Responsible Party
Project team meeting	*Daily*	*Meeting*	*Tech team / Project team*	*PM, Chair*
Stakeholder update	*Biweekly*	*Email / Website*	*Stakeholders (internal and external)*	*PM*
Leadership group update	*Biweekly*	*Meeting*	*Executive leadership*	*PM*
Regular newsletter / blog	*Weekly*	*Portion in newsletter / blog / etc.*	*Users / Community at large*	*Comms team lead*

Examples in *blue italics*.



TECHNOLOGY

Phase III: Preparing for Change

Activity 20: Winding down an Open-Source Software Program

Goals

1. Decide when a project or program is no longer useful
2. Understand how to disengage from a project or program
3. Determine what to do about code, repositories, websites, wikis, and other project or program assets

Prerequisites

None

Who Should Participate?

Program leadership (strategic), Program management (tactical)

Length

120+ minutes

Activity Instructions

Visit the Linux Foundation "Winding Down an Open-Source Project" Guide, linked below. From the Linux website:

This Open-Source Guide is designed to offer advice about how your enterprise and your development team can plan for the day when you are ready to end or move away from an unneeded open-source project. By shutting down the project gracefully or by transitioning it to others who can continue the work, your enterprise can responsibly oversee the life cycle of the effort. In this way, you can also set proper expectations for users, ensure that long-term project code dependencies are supported, and preserve your company's reputation within the open-source community as a responsible participant.

This guide will help you decide when a project is no longer useful, understand how to disengage from a project, and determine what to do about its code, repositories, websites, wikis, and other project assets as you head in a new direction.

https://www.linuxfoundation.org/en/resources/open-source-guides/winding-down-an-open-source-project/

Questions for Reflection after visiting the Linux Guide

1. Does your program have any of the trouble signs the guide discussed? Are the trouble signs confirmed with actual data on usage, number of contributions, or errors in the system?
2. If you are planning to disengage, have you decided whether to transfer the project, end your specific portion, or end the entire program? Have you thought through the pros, cons, and resources necessary for each option?
3. What is your plan for program assets, such as code, repositories, website, documentation, etc.?