ROADMAP MERGE: OVERVIEW

SUMMARY

You've already got too much to do. There are deliverables to make, deadlines to hit, multiple projects to manage. We get it. The point of this workshop isn't to add to the work you already have; it's to explore how to leverage that work by doing it within an open source community – or leverage an open source community to do your work; either way is great.

In order to do that, we've got to understand the deadlines and milestones you're already working under – and how they map into the timelines and rhythms of the open source community you've chosen. Maybe you'd like to use a feature coming in the next release; when will that be? Maybe your project could use a Test Day; when are those typically scheduled, and how far in advance are they planned? Maybe you're curious how the Infrastructure team prepares for a massive server-hitting launch or how the Translation team coordinates sprints before string freeze; where do you look to find out the dates you should be listening in on the conversation?

We'll show you a quick overview of what an open source project's release cycle looks like, then set you loose to hunt down the major milestones of your chosen project's current cycle so you can find out what's happening *right now*. Once you've got those dates, you'll use the Roadmap Merge Worksheets to lay out your existing work's timelines right next to your chosen project's timeline, then see if there are any natural opportunities for connecting or piggybacking in either direction.

LEARNING OBJECTIVES

At the end of this unit, you should be able to:

- Find and list the major milestones of an open source project's current release cycle.
- Draw a timeline of how the work you're already doing fits into a project's release and feature schedule.

VIDEO: 6 MONTHS IN 60 SECONDS – RELEASE CYCLE ROLEPLAY

Handout: roadmap-merge-video-handout.pdf and roadmap-merge-video-handout-blank.pdf (source: svg)

Fedora uses a time-based release cycle, which means that feature planning is done by picking a deadline, then asking the question “which features can be finished by this deadline?” In contrast, feature-based release cycles (for example, Debian) pick features they wish to see completed in a given release, then ask the question “when can these features be completed?” Both types of release cycles are common in the open source world.

Fedora's release cycle is 6 months long, which is an average release cycle length shared by major upstreams such as the GNOME Desktop. Other time-based projects may have release cycles as short as a few weeks or as long as a year or more. We're going to recap each month of the release cycle in 10 seconds, for a total of 60 seconds of explanation of the entire thing. Use the provided handout to keep track of

which teams within the project are doing what at what point in time – or feel free to take your own notes in whatever format you choose.

It's going to be fast. Don't worry if you don't get it all the first time; it's meant to give you a general overview, and we'll pause and walk together through each component, each milestone, and the various players – teams and individuals – onstage each month so you understand each component. We'll play it again full-speed at the end so you can see how all the pieces fit together.

RESOURCES: ROADMAP MERGE

Handout: roadmap-merge-resources.pdf (source: svg).

Now that you've seen a generic Fedora release cycle, it's time to hunt for the dates in the current and next release cycle of your chosen project. You might be looking at Fedora, you might not be; search engines are your friend.

Helpful search terms include:

- release cycle
- release schedule
- calendar
- roadmap
- deadlines
- feature process

If your project doesn't have a roadmap/schedule, call your instructor over. Think about why that might be the case – perhaps it's a new project, or a small one, or there is a schedule but it's undocumented – how could you find out?

The second part of this Resources page is for filling in information for the various teams you might encounter in your chosen project. Some teams you might encounter:

- Infrastructure (sometimes called Sysadmin)
- Websites (sometimes called Webmaster; may be part of Infrastructure or Marketing, or a separate team of its own)
- Marketing (sometimes called PR, occasionally merged with Ambassadors and/or News)
- Ambassadors (sometimes called Events or Outreach, occasionally merged with Marketing; may be further split geographically by country or region, or into individual local clubs/chapters.)
- News (not all projects have this; sometimes merged with Marketing)
- Design (sometimes called Art)
- Translation/Internationalization/Localization (Also known as i18n, t9n, and l10n - count the number of characters between the first and last letters of the word, and consider the difference between

British English and American English spelling, and the subsequent debates of how, exactly, to name these teams. These may be one team, or split into several teams.)

- QA (sometimes called Testing, Bug Squad, or Bugzappers)
- Documentation (sometimes called Docs)
- Packaging and distribution (may not exist for all projects; primarily relevant for distribution projects)
- Developers (sometimes called Coders; may not exist for all projects, particularly distributions)

Don't forget to look for SIGs (Special Interest Groups) – there may be regional or discipline-specific groups you're interested in. For instance, would you imagine the following to exist in Fedora?

- Multiple desktop groups – not just GNOME, but also KDE, XFCE, and others
- Musicians' SIG
- Education SIG (and OLPC/Sugar SIG)
- Finance SIG
- Campus Ambassadors (university-level)
- Summer Coding (Google Summer of Code and similar mentoring projects)
- Ham Radio
- Robotics

They all do. See if you can find them – and other groups like them.

WORKSHEET: ROADMAP MERGE

Handout: roadmap-merge-worksheet.pdf (source: svg). Optional: hand out sample completed worksheets.

This worksheet consists of a double timeline (the top is for the timeline of your current work, the bottom is for your chosen open source community) followed by a “mergepoint” section. A mergepoint is an opportunity you spot for one event happening at one time and one (online or real-world) location to “double-count” and score an Epic Win for both your work and open source projects.

Use as many worksheet pages as you need. Each worksheet page represents three months. We suggest using 2-4 sheets for best effect (6-12 months) and starting with the current month. Here's what you do:

- Write in the months on the timeline, so we can see what dates we're looking at.
- Plot in the major milestones, deadlines, and time-periods for your work project(s) on the top half of the timeline. Mark major holidays/vacations as well.
- Now look at the “resources” page you filled out previously. What milestones there are relevant to your work project? Draw them in on the bottom half of the timeline.
- (Mostly applicable to those working in Fedora.) Look at your “6 months in 60 seconds” notes. Are any teams doing activities you are interested in? Draw in when they happen on the bottom half of the timeline. For instance, “infrastructure freeze” may not be part of the feature process on the

release schedule, but if you're thinking about asking the project's sysadmins for advice on trying out a new tool, the frantic week before the freeze may not be the best time to ask.

- Look for a mergepoint. (If you find more than one, that's great!) Remember, a mergepoint is anything that could double-count for both your work and your open source project. For example, do you need to write a technical specification a week before feature proposals are due? Why not submit the same text for both? Is there an upcoming conference or hackathon? Maybe community members for your project would be interested in hearing you present your work. If you're stuck, look at the examples (if provided) or ask your instructor for help.

If you finish early, see if you can do the following:

- Describe at least 3 teams/contributor-roles within an open source project that your work will affect and/or be affected by, and where each team fits into the RACI model for your project.
- Adjust your work timeline in response to schedule slips in an upstream open source project. What sorts of dates and deadlines do your mergepoints depend on? What happens if they happen a week early? A week late? Two weeks late? How do you need to prepare and react (if at all) – and how can you make sure you'll know if schedule changes happen? Alternatively, is there a way you can modify your work to remove its dependency on that deadline, and what sorts of tradeoffs does that entail?
- Chase down more info on your “what needs to happen” -sections wave down someone in the classroom (an instructor or a fellow learner) and ask them to help you drill down on more details; pop on IRC and see if you can find a community member to answer your questions, search online and see if you can find documentation that answers your questions.
- If you're really feeling bold, start taking action on the “what needs to happen” list for your mergepoints!