Passing the Baton:

Succession planning for

FOSS leadership

All Things Open 2019

@vmbrasseur

🚨 START YOUR TIMER! 🚨

An origin story…

Me: FOSS for ~30 yrs, tech for 20

Open source business strategy

Seen a lot, good and bad

Part of that good/bad is the relatively recent conversation around FOSS sustainability. Many of us have been concerned about this for a lot longer than the popular conversation, and we haven't been focusing simply on money.

Several years ago I started talking to FOSS project maintainers about their communities and their governance, and things that they're doing to make things more sustainable, though we weren't using that now trendy word to talk about it.

This talk comes from what I learned in those conversations, some surveys I performed, and from my own business and community management experience.

YMMV

Succession planning can be a complicated thing at times. I will be presenting general tips for how to approach this process.

Not all of the recommendations will necessarily apply to your specific project or situation.

Some of the recommendations may apply better to large projects, others to small projects. It all depends on the project's requirements.

Take from the presentation what you need. Don't feel obligated to follow every suggestion to the letter.

Please save questions for the end

Brief history of FOSS

Let's set some context.

We've been sharing software for as long as there's BEEN software

But free/open software as a recognizable movement only started 40 years ago.

Let's quickly look at some of the many highlights of our history.

1970s & 1980s

- 1976: emacs
- 1983: GNU
- 1984: X
- 1985: FSF
- 1987: Perl

emacs: Moon & Steele > RMS > GNU > FSF

Perl: Created by Larry Wall

1990s

- 1991: Linux, Python
- 1993: NetBSD, FreeBSD
- 1995: PHP, GIMP, Ruby
- 1996: Apache, KDE
- 1997: GNOME
- 1998: "open source software" coined, OSI formed
- 1999: OpenOffice, Apache Software Foundation

Things started getting really exciting in the 1990s.

1998, Christine Peterson coins the term "open source software" and the Open Source Initiative comes into being to support OSS and the Open Source Definition

2000s and beyond

- 2003: Mozilla, Wikimedia
- 2004: Ubuntu
- 2005: git
- 2006: Software Freedom Conservancy
- 2007: OpenJDK
- 2008: Chromium, Android
- oh so many more things

A new century brought an explosion of free and open source software and organisations.

Things have really escalated in the past 10 years.

FOSS components have become the default selection for many companies

We've come of age

Mark Cuban: "Software has eaten the world."

Every tech journalist ever: "...and open source has eaten software."

There's no denying that the free and open source software movements have changed the face of technology & are here to stay.

...and we're starting to reach the scale and age where oral history doesn't cut it anymore.

40 years. 40. And we still have the founders of our movement in our midst. But for how long?

There are now ***millions*** of free and open source projects currently in existence...

According to the 2018 Octoverse, there are over 96 million repos on Github ALONE

This doesn't include the repos on GitLab, in the Apache or Eclipse universes, or any of the other code repository worlds

...we ***rely*** on them ***every day***...

Many of us could not do our jobs without these projects.

We couldn't run our companies. We couldn't serve our customers.

…so what happens when their leaders need to **move on**?

But now, 40 years on, we're overdue to be thinking about how we in FOSS will get by when our founders, our leaders, move on.

These aren't necessarily the Benevolent Dictators For Life of the world.

Every project has people in important roles. Every project needs to consider a backup plan for when those people leave the project for some reason.

Succession Planning

And that brings us to succession planning. Like I said earlier, I've done a lot of research into this in a FOSS context, including surveying and interviewing leaders and community members of open source projects and communities.

First I'd like to define Succession Planning for y'all so we're all on the same page.

Defined…

Succession planning is a process for identifying and developing new leaders who can replace old leaders when they leave, retire or die.

— Wikipedia

This is pretty standard for what you find in the literature about this subject.

I'm not a big fan of this definition. It's rather limiting. It implies only "leaders" are the ones who need backup & successors.

What is a "leader" for FOSS, anyway? When you stop to think about it, it gets complicated pretty quickly.

~~Leadership~~ Skills Pipeline

Training and mentoring people in the skills required to step into critical roles.

Literature focuses on high level leadership things like CEO, Founder, etc.

This is FOSS. There are other important roles beyond project founder, Benevolent Dictator for Life.

Any project role which is measured by bus factor is a role which can benefit from succession planning.

Important to note here that these are Skills, and as Skills they can be learned.

"A project's bus factor is a number equal to the number of team members who, if run over by a bus, would put the project in jeopardy."

But…why bother?

When I talk about this, some people say, "Meh, we've never done this in the past. Why should we think about it now?"

Well, for starters, "we've always done it this way" is a very dangerous statement. But there are some other reasons why you should think about succession planning…

Continuity

The most obvious reason. When someone leaves, what happens to the tasks they were performing?

They were keeping some plates spinning. When they leave, do those plates fall and break?

Succession planning helps ensure tasks continue to be performed and no one is left hanging.

Avoid a power vacuum

If a person leaves a role without a replacement, it can lead to a lot of confusion, delays, and possible political woes for the project.

Those political woes can be the most damaging to a group. Delays are more easily fixable than hurt feelings.

Creating a succession plan can help alleviate the very insecure and unstable time when someone in a vital role moves on.

Everyone knows who's next in line and why.

Project/organisation longevity

The thinking required for succession planning is the sort of thinking which contributes to project longevity.

The entire point of succession planning is that the group is taking the long view.

Ensuring continuity in leadership, culture, productivity also helps to ensure that the project will continue. It will evolve, but it will survive.

Reduce load/pressure on current leaders

Successor becomes backup

^ Leaders are more free to take vacations, etc.

^ Reduces burnout of project leaders & those in important roles

Talent development

We talk a lot about mentoring people to contribute code, but we also can mentor people to become successors for critical roles

This helps everyone: leaders get successors, project/organisation gets continuity, successors get career development and experience.

Inspiration/motivation for new members

It can be very motivational for new community members to see a succession plan in action.

"There is a path to and plan for leadership positions here. That could be me some day. I would like to stick around."

Also instills confidence that the project has its shit together.

Diversity of thoughts/Get out of a rut

Succession plans are a great opportunity to bring new people, new ideas into the leadership roles of a project.

Studies show that diverse leadership teams are more effective and the projects they lead are more innovative.

Use your succession planning as an opportunity to open the door to diversity.

Truly meritocratic

Disclosure: I have a problem with this whole "meritocracy" bullshit.

Many projects in FOSS are proud of their "meritocracy." Conceptually, it's a good idea.

^ Unfortunately what passes for meritocracy typically translates to hostility toward new contributors and echo chambers.

If you can't express what "merit" is beyond "I'll know it when I see it" then you have not thought things through very well.

A meritocracy without a mentoring program and healthy goverance structure is just an excuse to practice subjective discrimination. It's a way for bullies to hide behind unexpressed biases.

IMO, you cannot claim to value "merit" if you do not take the time to train people to earn that merit or even teach them what it is that earns them that merit.

A well-executed succession plan can help with this.

A well-executed succession plan is open, honest, documented. There are very few opportunities to hide behind weasel words like "meritocracy."

Benefits of FOSS Succession Planning

Continuity	Talent development
Avoid a power vacuum	Inspiration/motivation for new members
Project/organisation longevity	Diversity of thoughts/Get out of a rut
Reduce pressure/load on current leaders	Truly meritocratic

As you can see, there are a lot of really great things which can come from having a succession plan.

It's not a panacea. There will still be problems. But there are a lot of really worthwhile benefits here.

Despite that, very few FOSS projects or organisations put much thought into it.

Why?

Why it doesn't happen

From my research, I've found the following reasons for why projects don't engage in succession planning:

Too busy

A lot of people recognized succession planning as a problem in their project but just "hadn't gotten around to it" because there's "always something more important to work on."

While I can understand this, I really can, I personally think this is more a problem with prioritization than with time availability.

It's likely these people may not yet realise how bad it can be not to have a succession plan when you need one. And that's why I'm talking to you today.

Don't think of it

Some are just so busy & preoccupied that they haven't even thought about, "Hey, what would happen if Sarah left the project?" It just never occurs to them. I mean, Sarah's always there when we need her, right? Always.

Don't **want** to think of it

Succession planning is often, like estate planning, associated with negative feelings like loss and can make people address their own mortality. Some people aren't comfortable with this and avoid it.

Attitude of current leaders

Current leaders don't want to recognize that they're replaceable or to consider that they might give up their power and influence.

Failure to recognise or admit this can set the project up for failure in the long run.

This doesn't happen often but it does happen and is worth mentioning.

Don't know where to start

Some know it should happen, are willing to carve out the time,

but don't know how to do it.

How to develop FOSS leaders of the future

OK, so how do you do this?

How do you even get started?

DON'T PANIC

I'm about to throw a lot at you.

DO NOT feel you have to do it all or do it immediately.

This is a process. Again, the thing about succession planning is you're taking the long view. That means it can & should take time.

So don't worry if you don't feel you're making progress. If you're working on it at all, you ARE making progress. Congratulations.

Also, I'll give suggestions not just for current leaders, but also for those who would like to move into critical roles.

Current leaders

So, if you're currently in a critical role in a project, what can you do to help cultivate people to take your position when you move on?

Step 0:
Don't work in isolation;
work on this together & publicly

Before you start, remember that you're in an open community and do all of your work in the open

Solicit feedback from the community and keep them posted on what's happening and why

Step 1:
Identify critical roles

Step one is to identify those roles in your project which are fairly critical.

Only you & your project can define what qualifies as "critical" to you.

Role != Person

Note: While it helps to start by looking at the people on the team, it's not always to correct that each person is performing only one role

Step 2:
Identify duties & responsibilities for each role

Now, once you've identified those roles, be very honest with yourself and your community about the duties of each role.

List what you think the role should do

but also list those duties it ACTUALLY performs. The second list will probably be longer.

Be honest about the duties a person is performing and how they might belong to multiple roles

Refactor large roles

If a role does a LOT, then you should break it up into multiple smaller roles

Refactor: redistribute roles amongst other people -> Reduce bus factor right there

Refactoring also makes it much less intimidating for someone new to take on the role. The new role will be smaller and easier to step into.

Consider limiting tenure for a role

Ensure that it will be handed off

Give current role-holder a light at the end of the tunnel

Ensure there will be a body of people familiar with the role (past role holders)

Step 3:
Knowledge transfer

Once you've identified & refactored the roles, a lot of the work consists of knowledge transfer

While this should be an on-going process, but you'll probabsly need a big initial brain dump once you've collected the initial information it should just be a matter of sharing it and maintaining incremental updates.

So what knowledge should you be sharing?

Procedures & processes

All those duties don't occur in a vacuum. How does one perform them?

Some may be self-evident. Others may not.

But remember: you have a different perspective. Self-evident to you isn't to others.

So please be explicit and transfer all knowledge about all procedures & processes.

List of accounts: What accounts exist out there? Travis? Twitter?

List of contacts: Do you have meetups? Who are the people who usually help you with meeting space, sponsorship?

Credentials

No one person should ever have access to credentials for services used by the project.

This is one of the most common issues faced by projects when faced with a succession situation

People walk away with the keys and no one else has copies

I've also seen this power used for evil (shutting folks out, embezzling, etc)

Always use role email accounts sent to multiple people, not individual/personal accounts.

Provides the context necessary to make the best decisions in the operation of duties.

Inside jokes

Knowledge Transfer

- Procedures & processes
- Resources (online, people, and otherwise)
- Credentials
- Project history
- Inside jokes

Step 4:

DOCUMENT

ALL

THESE

THINGS

Now, you've learned & discussed all these things.

I mean, this has probably taken a LOT of a lot of peoples' time, amiright?

FOR PETE'S SAKE, WRITE THEM DOWN. Don't lose this information.

The How & Where doesn't matter nearly as much as the Whether.

Capture this information for posterity. This is a talk on succession planning. For crying out loud, think of future generations of project leaders and write this stuff down now.

But consider tl;dr docs

Can be helpful to provide an overview doc (more bite-sized)

A'la an abstract for a research paper

New leaders

So don't think this only has to be the purview of those already in critical roles.

Those of you who are interested in contributing at a higher level can do a lot to help this process.

Look for opportunities to learn & contribute

For starters, look around. Where can you help those in critical roles?

Are there opportunities for you to learn more about the leadership and operation of the project?

Ask to chip in. Volunteer for the small, possibly annoying, but definitely important tasks.

Volunteering for the grunt work is a great way to apprentice your way into a leadership position.

Shadow those in critical roles

Another way to learn and apprentice your way into one of these critical roles is to ask whether you might shadow one of these role-holders.

See how the role is performed. What is the process & procedure?

See what's actually involved with the role.

Confirm whether you actually want to perform the role.

Ask for mentorship

We who mentor...we're human & we're busy. We don't always have the mental cycles to recognise that mentoring it needed or wanted.

We're usually happy to provide it, but sometimes we need a nudge. Please ask!

Related to that, when you see someone performing critical duties, ask whether they'd be willing/able to teach you about it.

Ask for feedback on tasks you perform.

Ask for tasks you might perform to assist them.

Learn the history of the project/organisation

Sit around the campfire with the project elders & listen to them spin yarns of the Good Old Days

Extra points: write what you learn and share it for the benefit of your entire community

Examples of projects working on this

Was going to give examples of projects doing this badly but won't for 2 reasons:

1. I don't believe in naming-shaming for stuff like this.

^ 2. My research shows I don't need to give examples of projects handling this badly because everyone already knows some.

Instead, I'll give you examples of projects doing this well.

Please note: This is not an exhaustive list.

Exercism

"The goal, of course, is to have at least three active maintainers for every single language track. This way we could put some procedures into place to mentor new contributors, nominate new maintainers, and to roll off the project without any sort of guilt when you no longer want to maintain a project."

— https://exercism.io

Complete exercises to gain fluency in programming languages

Over 50 programming languages, over 3000 exercises

Rated their tracks by how maintained they were.

Making very active efforts to make sure each track has not one, but multiple maintainers

Succession planning in action: always someone available to step in when one of the maintainers isn't available.

Gives maintainers breathing room & a chance to take a break when they need it.

Vox Pupuli

"One of the benefits we hope to achieve is that by a shared ownership of modules we no longer end up in situations where the original maintainer has moved on and a forest of disparate forks try to fill the void."

— https://voxpupuli.org

Ensure continued development of Puppet modules

176 repositories of Puppet modules

VM (Vicky) Brasseur

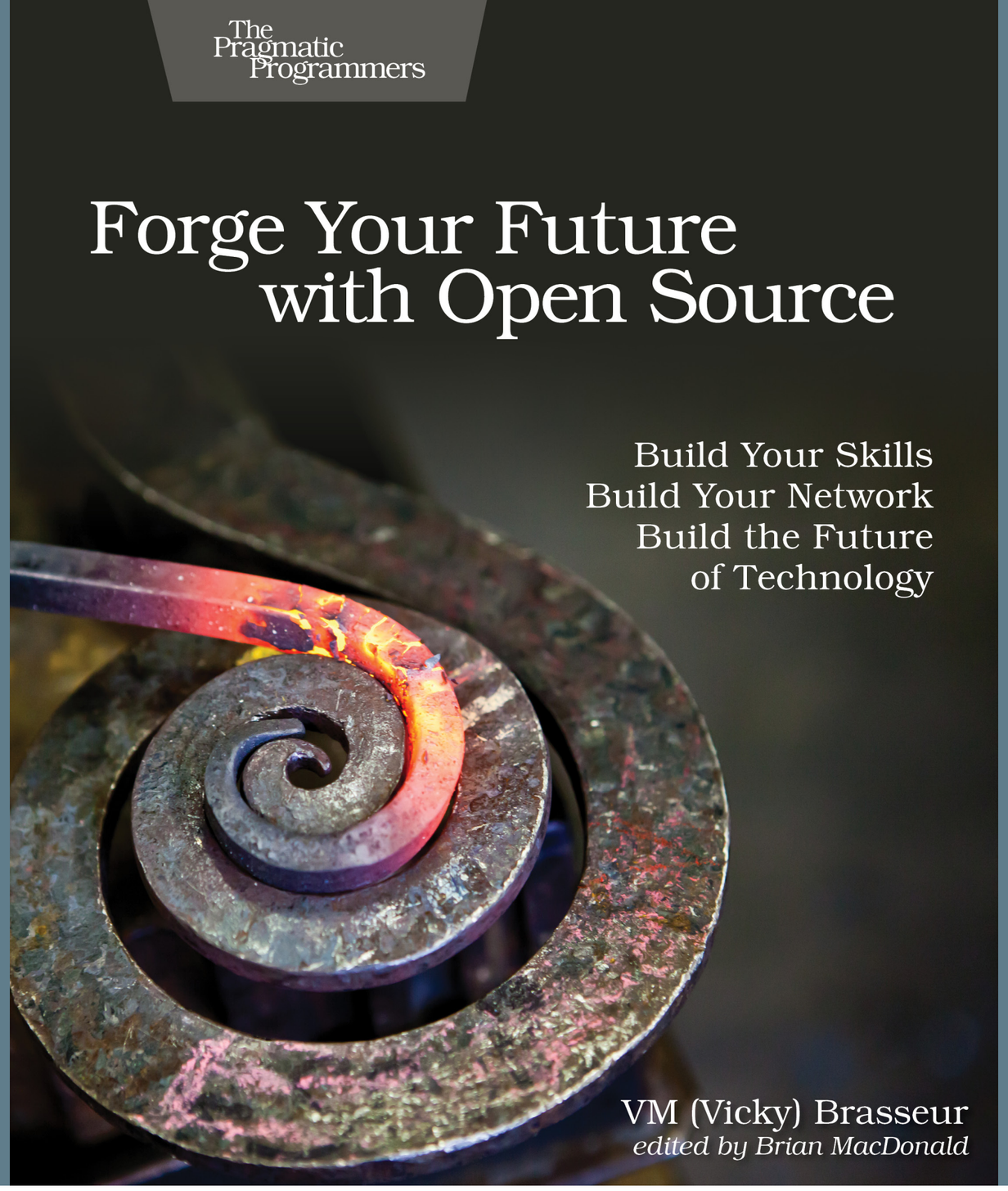
@vmbrasseur

• • • • •

Slides: **https://archive.org/details/ato2019-successionplanning**

• • • • •

Book: **https://fossforge.com**



I am VM Brasseur, Director of Open Source Strategy for Juniper Networks

I am also the author of this, the first and only book about how to contribute to free and open source software projects. I'll be doing a book signing at 11:40 right after this session.

You can find me here at Twitter

You can find these slides already here on Internet Archive.

You can the book at this URL.

Thanks to the All Things Open team for having me.

And thanks to you for being here.

Now, any questions but not comments disguised as questions?