

FROM STORMING
TO PERFORMING:

GROWING YOUR PROJECT'S CONTRIBUTOR EXPERIENCE

Microsoft Azure

Matt Butcher // Principal Software Engineer

Karen Chu // Community PM



+

.

WHO ARE WE?



Matt Butcher

- Principal Software Engineer // Microsoft Azure
- DeisLabs
 - Open-source lab at MSFT
 - Created stuff like Helm, Brigade, Draft, Krustlet, CNAB
- Co-author of *The Illustrated Children's Guide to Kubernetes* book series
- Twitter: @technosophos



Karen Chu

- Community Program Manager // Microsoft Azure
- CNCF Ambassador, Kubernetes Code of Conduct Committee member, TAG Contributor Strategy
- Community manager for DeisLabs
- Co-author of The Illustrated Children's Guide to Kubernetes book series
- Twitter: @karenhchu

+ .
o

YOU'VE GOT YOUR NEW OPEN-SOURCE PROJECT... NOW WHAT?

+ .
o

Form // Storm // Norm // Perform

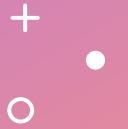
In 1965, a psychologist named Bruce Tuckman reviewed the literature on group dynamics and came to the conclusion that groups go through a common sequence that he called "form, storm, norm, and perform".

We think that model can be applied to open-source projects.

Tuckman, Bruce. 1965. "Development Sequence in Small Groups." Psychological Bulletin 63(6).

FORMING

TACKLING FIRST
TASKS TOGETHER



FORMING: Tackling first tasks together

+

o

Contributing code

- Dealing with pull requests and issues

Having a website

- Branding/identity
- Netlify/Github pages (quick and easy ways to get your website up)

Communications

- Clear places to hold conversations and find the community
- Slack, Discord, Gitter, Twitter

Maintaining docs

- You should have a quick start guide and reference information

STORMING

GAINING TRUST &
SORTING THINGS OUT

STORMING:

Gaining trust & sorting things out

Disputes are unavoidable	Code of conduct	Governance	Coding standards	Contributors are not employees/coworkers
<ul style="list-style-type: none">• Be prepared to work through conflict amongst maintainers and community members	<ul style="list-style-type: none">• You should have one• This protects the community from becoming about the wrong things• It creates a safe and welcoming space for the full breadth of people who want to use your project	<ul style="list-style-type: none">• Make it clear how decisions are made in your project and who makes those decisions• Document it	<ul style="list-style-type: none">• A contributor doc should explain base-level information about formatting, idioms, terminology, etc. (to help set expectations)	<ul style="list-style-type: none">• You can't expect to be able to tell people what to do (they have their own goals)

+ · NORMING

o

SHARING RESPONSIBILITIES

NORMING:

Sharing responsibility



Issue management

Triaging

How do we do our labels well?

Making sure the issue queue is reviewed



Delegating work

A growing project needs core maintainers

Find volunteers to lead

Make it clear how somebody can claim a feature

Who decides what work gets delegated and how it's delegated?



Standardizing communication channels

Avoid the information firehouse

Use security mailing list for communications

Use maintainers-only list for voting

Official decisions need to be documented

Outgrowing single Slack channels

PERFORMING OPTIMIZING FOR THE LONG HAUL

PERFORMING:

Optimizing for the long haul



Retaining maintainers

- Avoiding burnout
- Incentivizing people to continue onward

Contributor turnover & recruiting

- Very few people will stay for the entire life of the project
- Assign responsibilities to more than one person
- Be proactive in getting new contributors
- Converting contributors to maintainers

Emeritus status

- Gives new people an opportunity to step up
- Asks people to step down vs people volunteering to step down
- Allows people to point out their contributions to the project, even after they've left
- Formally triggers offboarding (knowledge transfer, permission changes, clear messaging around stepping down)

CONCLUSION

+

•

○

+

•

○

Conclusion

- There are limits to the model
- Just because you get to "performing" doesn't mean you will stay there
- Sometimes you may need to check back in on the "norming" or even the "forming" stage
 - New major releases of the software sometimes shake up the team
 - New maintainers can change dynamics
- Because your community is always changing, "storming" can happen by surprise
- Years later, the psychologist Bruce Tuckman added one more stage: "adjourning"

QUESTIONS?



THANK YOU

+ .
o