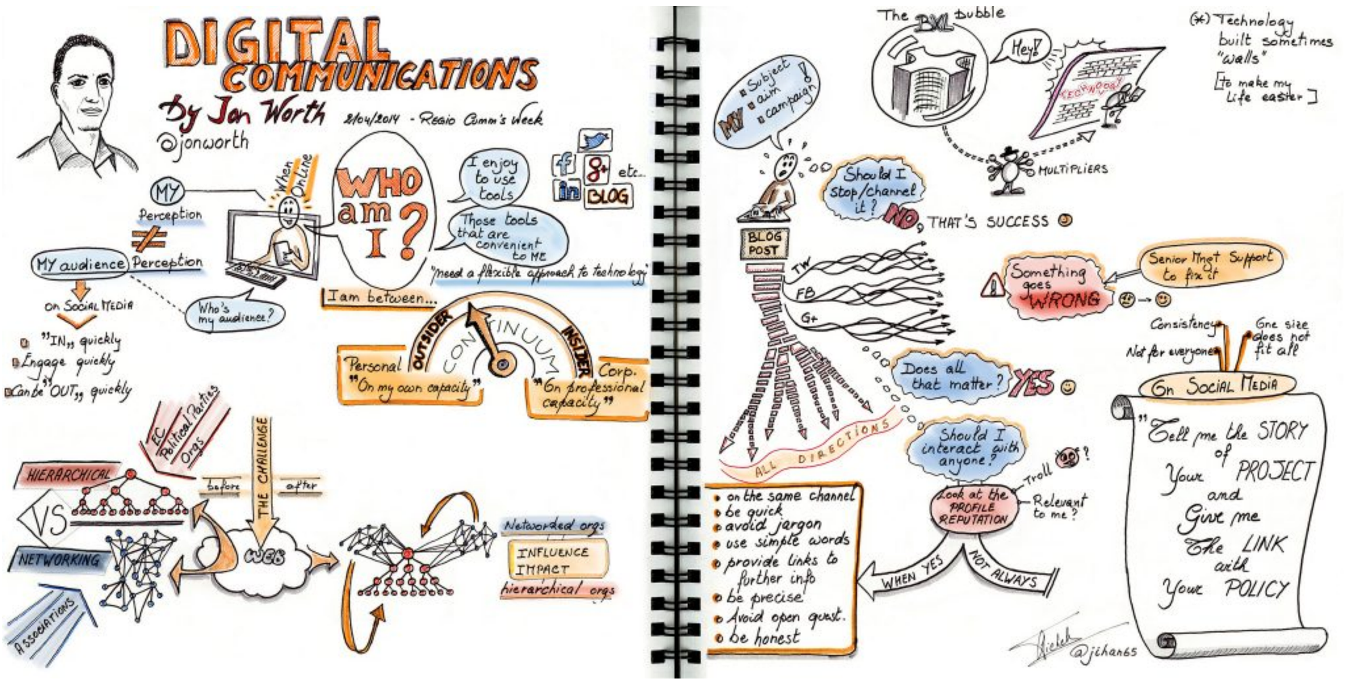
Effectively using Discourse together with group chat

Erlend Sogge Heggen April 10, 2018 use cases



Most modern businesses and organisations today are using some kind of team chat application. The usual suspects are Slack, HipChat, Discord, Mattermost, Rocket Chat, Riot, and Gitter, to name a few.

While chat is **immediate** and primarily **synchronous**, communication in Discourse is **gradual** and **asynchronous**. We've seen far too many community managers treat these two modes of communication as competitors. Quite on the contrary, chat and forum communities can complement one another beautifully, and we aim to show exactly how by breaking it down into three different levels of understanding.

1. **Ephemeral vs Permanent**
   What is the difference between a chat and a forum community
2. **Strengths and Weaknesses**
   When to use which tool
3. **Complementary Workflows**
   How to use both tools together most effectively

Ephemeral vs Permanent

The discussion that takes place in a chatroom is best described as ephemeral, meaning:

> Something which lasts for a short period of time.

While a chat platform might keep a searchable record of all your conversations, the highly unstructured nature of chat makes it unsuitable as a long-term storage of knowledge.

In direct contrast to the inherently ephemeral nature of chat, forum discussions are permanent:

> 1. Without end, eternal. *Nothing in this world is truly permanent.*
> 2. Lasting for an indefinitely long time. *The countries are now locked in a permanent state of conflict.*

One mode of discussion is not better than the other. They both have unique benefits that go hand in hand with best-practice use cases.

Strengths and weaknesses

Chat is an essential communication tool, but when used excessively it can have a negative influence on community health. Whereas with forum communities, perhaps the most common misstep is to start one before your project has the necessary momentum, and you end up with a ghost town.

At Discourse we like to think of chat and forum as your extended team's collective "short term" and "long term" memory, respectively.

Group chat is great for…

1. **Minimum viable communities**
   Get two people in a chat room together and you've got yourself the beginnings of a healthy community. As long as there's some chatter on a regular basis, the room will come off as lively and inviting to other prospective participants. This is a great onboarding strategy in the early days of a community, but there is a hard limit on how far it can scale. Doing things that don't scale can be a winning strategy for startups and burgeoning communities alike; the key is knowing when you've outgrown your initial growth strategy.
2. **Real-time resolutions**
   "where's our invoice template again?"
   "I need fresh eyes on this weird query"
   "welp, I think I just broke something"
3. **Urgent notifications**
   Servers are on fire!
   An `urgent` email was just received!

4. **Socialising**
   While social chatter is a rare occurrence on a forum, this is the norm in chat, and it happens in *every* room, not just a designated `#off-topic` channel. Give it a little time and you won't be able to avoid it: *it's about to get personal*.There's something about being present together in the same slice of time, the *here and now*, that lends itself to Getting Real. Getting personal on a forum feels more like broadcasting, made even weirder by metrics (Likes, Replies, View) that can unintentionally imply that Bob's piece of personal news didn't "perform" as well as someone else's. Chat imposes much less scrutiny on your shared content.

You'll find this list to be nearly identical with that of Jason Fried's excellent writeup on group chat on behalf of Basecamp. While eerily similar, we really did come upon these findings independently! The proof happens to be permanently recorded on meta.discourse.org and in many other communities where we discussed this.

Forum discussion is great for…

1. **All-inclusive dialogue**

   The asynchronous nature of a forum community effectively lowers the bar about as far down as it can go. You'll get a much greater diversity of input if you solicit feedback from anyone who's available *some time* in the next 24, 72 or 168 hours as opposed to *right now*.

   Another little discussed benefit of "slow" asynchronous conversations (fun fact: very few things in life, especially in business, are actually URGENT) is that it encourages walking away from the discussion for a while, which is scientifically proven to improve critical thinking.

2. **Communities of scale**

   Similar to what version control did for code and wikis did for encyclopaedias, community platforms like Discourse have long since solved the "too many chefs" problem for discussion at scale. Hundreds or even thousands of people can discuss an equal amount of topics simultaneously on Discourse because (1) discussions are broken up into logical topic blobs and (2) long-form input is strongly encouraged over rapid-fire back-and-forth debating.

   > After 4 years of running a (11,000 member) public Slack community for @TryGhost — today we've decided to shut the whole thing down for good.
   >
   > I've got a few interesting observations to share from the experience, and what we're moving to now instead.
   >
   > And so beginneth the thread:

> — John O'Nolan (@JohnONolan) April 2, 2018

3. **Knowledge storage & distribution**

   The permanence of a Discourse topic makes it an excellent storage of knowledge. Some will argue that forum posts become outdated, but we've found that when a particular solution stops working then users will promptly resume the topic discussion until it arrives at a satisfactory solution once more.

   This is further helped by

   - Search-friendly content
   - Discoverable content, helped by Categories, Titles, Participants, Top rankings and strictly linear discussions with minimal digressions and noice.
   - Extra exposure for content with many Likes
   - Marking solutions as the official answer
   - Collaborative editing with full revision history

4. **Civilized discussion**

   Most current group chats have very limited moderation controls. And while they may eventually catch up, the experience will always be sub-par simply because it's real-time. If you hold someone's comment for moderation in chat, that's incredibly frustrating because you're expecting live discussion. On a forum on the other hand the expectation is that you'll get a reply within a few hours or even days after posting, so if your post gets flagged? No biggie, you can wait.

Complementary workflows

So let's assume you've already made the wise decision to use both chat and Discourse for your community *and* you understand when to use which tool. But how do you best use them together?

Continually enforce your communication policies

It's important that your project's community leaders (which should ideally translate to *every major contributor to your project*) actively enforce the norms that you're striving for. Lead by example and politely nudge newcomers in the right direction when they're asking for guidance in the wrong place or in the wrong way.

> **New user:** How can I do X?
> **Helpful user:** Good question. Please re-post this to our public forum so that any answers you receive can be searched for and read by anyone else who might be asking the same thing.

We’ve heard of some communities utilising bots with some success for this type of best-practice enforcement. If anyone has any such stories to share with us, please send us a mail at team@discourse.org!

Feed highlights from Discourse into chat

Important discussions in Discourse should be fed into your chat for extra exposure. There are many ways to do this:

- Use our standard chat integrations.
- Set up a custom integration with a using our API, webhooks or RSS feeds.
- Share the content manually.

It’s important to note that you shouldn’t be mindlessly feeding all of your Discourse discussions into your chat. Make sure you only cross-post the major highlights, i.e. just a few select categories or tags and the occasional manual curation.

Let chat conversations run their course, then post a summary

When a conversation takes place in chat that is related to an existing thread on Discourse, point out the link to the Discourse thread in chat. Don’t try to move the conversation over while it’s in flight though – that’s often disruptive. Instead, if anything *new* came out of the chat, encourage the primary chat participants to summarize it in a follow-up post on the Discourse topic.

Similarly, any chat content with lasting value to other community members should be exported over to Discourse once the initial burst of chatter comes to an end. Examples include:

- team members walking through a problem together and arriving at a solution
- deep conversation that is sure to continue for multiple days
- new user who asked a simple yet frequently asked question
- minutes of a meeting

Specifically for Slack, we have a very handy tool to help with this.

Export transcript from Slack chat

from **Erlend Sogge Heggen**

00:46

`/discourse post 20` will create a draft topic containing the last 20 slack messages. See this how-to for more info.

> This is currently only possible with our Slack integration. We welcome other group chat services to talk to us about how to make this possible for their platform. Please contact team@discourse.org

For some further reading, check these out.

Must-reads:

- Is group chat making you sweat?
- Slack, I'm breaking up with you

Also interesting:

- Discourse vs Slack
- Ghost moves from Slack to Discourse
- Should Ember better define its use of Slack?
- Curing our Slack addiction
- How to make Slack slightly less bad for you

Continue the discussion meta.discourse.org