Originated: Scrum Coaching Retreat, Kyiv
Updated: 31.05.2017

COMMUNITIES OF PRACTICE. A LIGHTWEIGHT COOKBOOK

Suggest corrections and contribute:
by opening this paper as a Google Drive Document.

Initial contributors:
Peter Hall, Alexey Krivitsky, Yuriy Malyi, Nataliya Trenina, Anton Vidishchev.

Best communities of practice we know:

1. … **are pain-driven**, created when needed
2. … are based around **volunteers**
3. … **run by the participants** for the participants
4. … **are creating enough long-term value** to balance short-term needs of the members
5. … **get dissolved** when the need is gone

We have also seen zombie-communities:

- **Run top-down** - management decides a new community is needed
- **No real need** - participants have no particular pain/need to get addressed (usually caused by the "top-down")
- **No real value** - the topics raised have no real value for the participants, the things being discussed won't change anything in the workplace (usually caused by the "top-down")
- **Obsolete** - there used to be a real need and a community around it; now there need has gone but the community is still there by the power of inertia

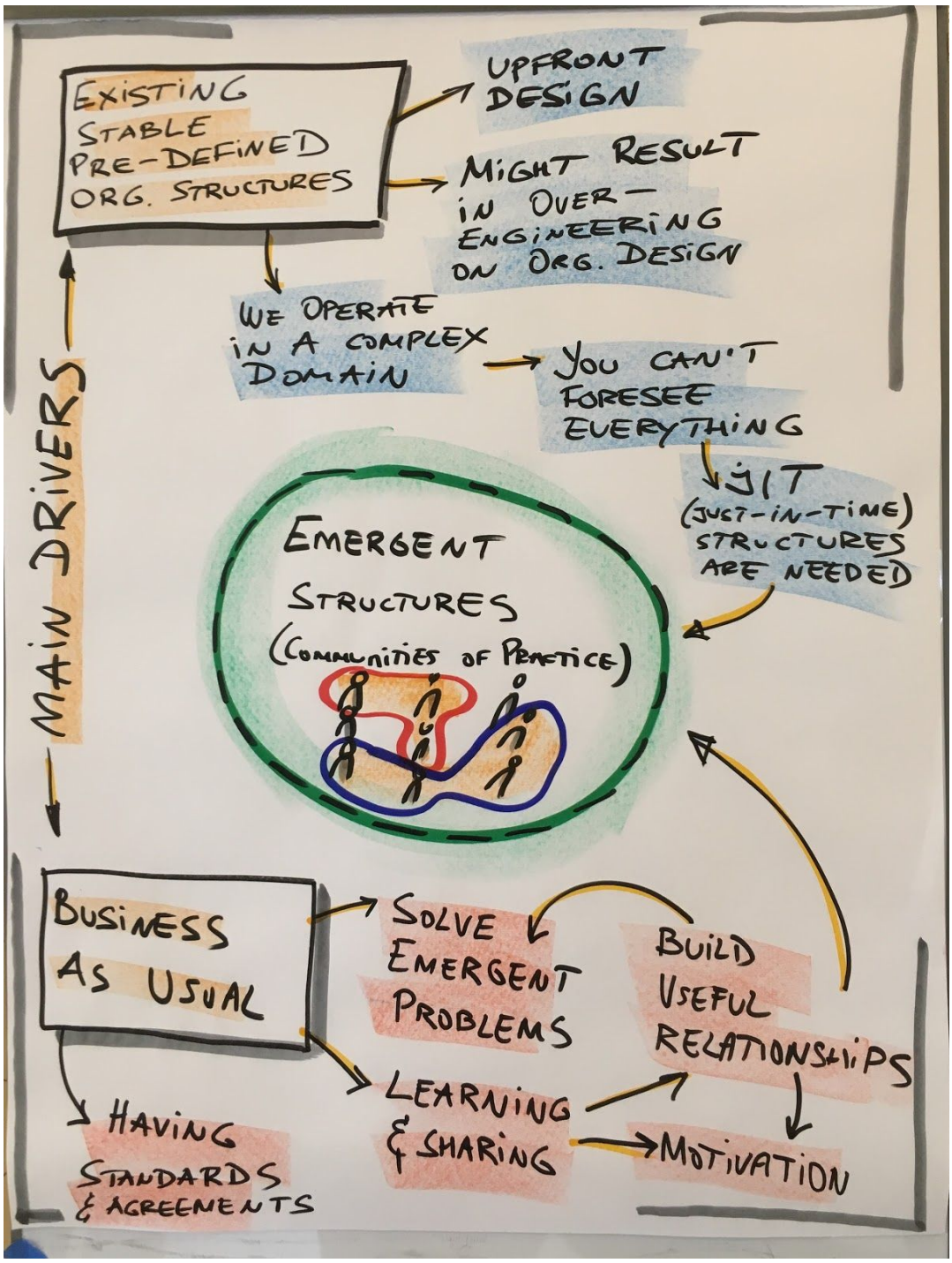
Why communities of practice (CoP)?

There seems to be at least two key drivers for emergence of new dynamic structures like communities of practice:

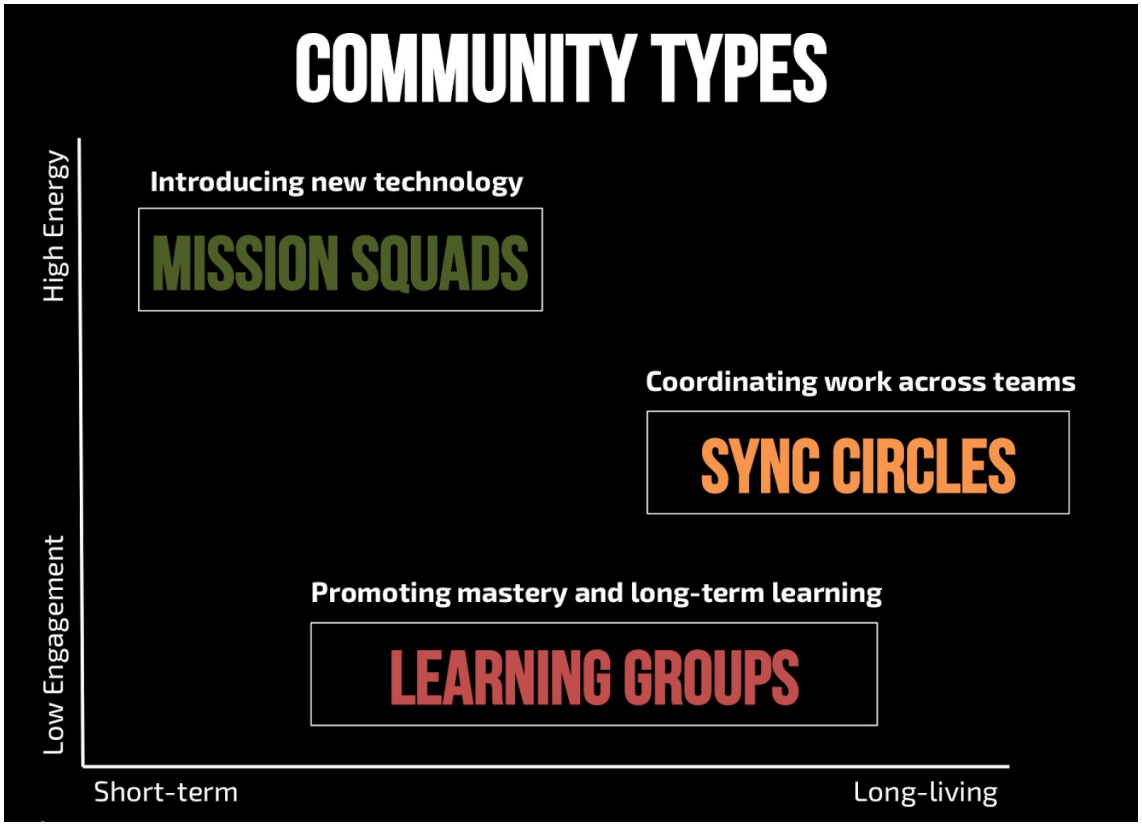
Because done upfront, standard predefined organizational structures can't solve all problems emerging from a complex domain, like product development. So the need for more lightweight, dynamic, just-in-time structures arises.

Also we know that knowledge workers are constantly facing with problems that the existing organizational structures are not designed to solve, so they are in constant search for hacking around an existing org. chart to get things done.

We also know that mastery (learning) as long as relatedness (sharing, building relationships) are strong motivational drivers for most of us.


EXISTING STABLE PRE-DEFINED ORG. STRUCTURES
UPFRONT DESIGN
MIGHT RESULT IN OVER-ENGINEERING ON ORG. DESIGN
WE OPERATE IN A COMPLEX DOMAIN
YOU CAN'T FORESEE EVERYTHING
JIT (JUST-IN-TIME) STRUCTURES ARE NEEDED
EMERGENT STRUCTURES (COMMUNITIES OF PRACTICE)
MAIN DRIVERS
BUSINESS AS USUAL
SOLVE EMERGENT PROBLEMS
BUILD USEFUL RELATIONSHIPS
LEARNING & SHARING
MOTIVATION
HAVING STANDARDS & AGREEMENTS

We've seen several types of communities:



A. Mission Squads *

main goal: introduce a new thing, make a change e.g. "continuous delivery chapter"

B. Sync Circles **

main goal: coordinate work across teams e.g. "android app release weekly sync"

C. Learning Groups

main goal: promote a certain skill e.g. "functional languages chapter

* Mission Squads are the most active, pragmatic and problem-solution oriented communities. They are also the ones which is clear when are not needed. So if we were to choose only one of the three mentioned community types - we would stick with this one only. In fact, question presence of the other ones.

** Some may argue if a Sync Circle is a community of practice at all. We think it is, provided it satisfies the 5-point list "Best communities of practice we know" above (i.e. provides long-term value for participants on voluntary bases).

Communities we've seen working well:

A. iOS developers from different full-stack teams meeting bi-weekly to show and discuss code snippets, discuss library updates, share key challenges in feature development. [Learning Group]

B. Project managers with a strong knowledgeable leader is meeting weekly for more than a year to watch learning videos and discuss good management practices. [Learning Group]

C. Android developers from different teams contributing to a single releasable app meet for 30 minutes on weekly basis to coordinate on the upcoming app release, solve emerging issues and improve the overall development practices used [Sync Circle]

D. Java developers get formed to deal with an upcoming Java update. Once the update is done, the group dissolves (until the next update). [Mission Squad]

Communities that had hard time:

- … **were not formed based on a real needs** of workers, but rather on a wishful thinking of management: "they shall be a community" (e.g. we know of a struggling community of Scrum Masters who worked in very different parts of an organization, didn't have shared goals or a list of impediments, and were not empowered to change things on a systemic level - so why bother contributing to a community?)

- … **were not able to provide its members enough long-term value** as people were only interested in gaining short-term results (e.g. we've heard of struggling component communities, apparently developers were touching a lot of components to implement necessary product features, so the costs of being a part of a long-term component community was not well-balanced with a short-term gains developers were mainly interested in)

Watch out for community abuse!

Power and reporting structures wrapped in communities.

People who might have lost their power and influence as a result of organizational transformation (e.g. members of component teams are now moved to newly created cross-functional feature teams thus leaving the ex-manager lonely and with no subordinates) are likely to create communities to try to regain their power:

- Example: QA engineers from cross-functional who had to "do their time" to serve their ex-boss (QA manager of long-gone QA

department) attending weekly status meetings to keep their boss informed, updated and self-important.

Hidden feature requests and pet projects

Similar to the above: some people with hidden agendas might abuse the idea of a community of practice to let people (maybe subordinates in the past) work on something that is not aligned with an overall strategy of the organization.

- Example: a "framework" or "platform" communities working off a secret technical backlog of a tech lead, for instance extracting an internal API to be exposed an open-source library for no real reason but self-service.

Imposing top-bottom decisions

Lacking ways to influence the decision making process and the key decision-makers and strategists (a group of Product Owners), some people (an architect) would try to sneak in a community to impose some secret agendas

- An "architecture" community led by an architect to impose standards on to development teams with no need articulated bottom-up.

A role of a community leader

We've realized all strong value-adding communities have (or used to have) a strong visionary and a collaborator who kickstarted the movement.

So anyone who sees a need of forming a community is a potential community leader.

And it can be you. Yes - you.

What leaders of well-functioning community do:

- They are very clear at articulating the need for a community to emerge and welcome volunteers.
    - So before going big and loud, they talk to several potential community members to check with them if the need is in fact real.
    - Or alternatively: they send out an invitation (with a clearly articulated need for collaboration) and see who comes and stays to work (after the initial curiosity settles down).
- They usually facilitate face-to-face decision- and work-oriented community meet-ups and help the group jell and start making real progress.
- They propose and then help create environment (tooling) supporting the ongoing community work.
- As time passes they are stepping back to foster decentralization of community ownership and leadership (new leaders might emerge at this point).
- They have a clear idea when the community is no longer needed and are constantly checking with the members if enough value is being produced vs. time/effort spent.

Tips and Tricks:

A community leader

- **Keeps participation voluntary** - community work is not for all, it is just for the ones who care enough, so never force participation.
- **Finds suitable tools for communications** and online/offline work (focuses on simplest tools possible, e.g. wikis, google docs)
- **Tries to keep the heart beat with activity cadence**
    - For learning group: lightweight meet-up with close to none preparation work well - consider webcasts and video lessons as a reason to gather
- **Makes goals and plans visible** and easy to contribute
- **Explores the ways for community to operate without her**

An upper manager / formal leaders of a company with communities:

- **Make sure people are aware that forming communities is welcome** and without any permission, and that communities are the first-class citizens as any other daily routines
    - Running once in awhile an overall company-wide community day might be a good idea to accomplish several goals at once:
        - 1) let the employees learn and explore the idea of communities;
        - 2) create environment for people to meet, talk and solve immediate issues at hand - to avoid forming unnecessary communities;
        - 3) facilitate forming of real communities based on long-term vision and actual needs;
- **Advocate for community autonomy and self-organization** (instead of pushing their process and content agendas onto communities)
    - Creating too many communities at once (just because management thinks they need to exist) is a common anti-pattern we see here and there.

  - Copying the Spotify model* (with its guilds and chapter) just because it sounds and looks cool might not be a good idea after all.
- **Serve the community when asked**
  - If there's a budget for a community - let the community decide what they want to do with it (instead of pushing external incentives onto community members that would undermine the value people are actually getting from the community work)

* The "Spotify model" is well-described by Henrik Kniberg: http://blog.crisp.se/2012/11/14/henrikkniberg/scaling-agile-at-spotify

Ciao! And bon appetit tasting emerging organizational structures.

The “emerging koalas” team from the Scrum Coaching Retreat, Kyiv, May 2017:
Peter Hall, Alexey Krivitsky, Yuriy Malyi, Nataliya Trenina,
Anton Vidishchev.