The Lead-Squad Protocol

Managing Work Between Squads

What's this?

Squads are designed to be as autonomous as possible to quickly succeed with a specific mission ("Increase mobile digital sales", "Provide the best identity service", etc.). Nevertheless, some objectives or opportunities may require several squads to work together. In these cases, the autonomous squads must find smart ways to coordinate work between themselves. This is the intention of the Lead-Squad Protocol: a set of simple rules that allows a **Lead Squad** to coordinate the work of several **Contributing Squads**.

Who is this for?

For *any* squad driving development work that requires the collaboration of other squads to succeed. It could be a small function that requires two squads to coordinate their work, or it could be to a major initiative ("*Make 5G available to all customers*") that requires 20 squads.

Eight pages document?! Do we really need to do all of this?
Well, it all depends on how big your development effort is. If you need to coordinate the work of 10 squads and 20 stakeholders under 3 months, then – Yes! – it needs to be that complex. If you only need to coordinate with another squad for a couple of iterations, then you do not need to go all in. These guidelines are here to help you, so following these will generally make you succeed better.

The Phases

The "squad-leading process" can be roughly divided into seven phases:

0- **Discovery** (usually done on beforehand) – where someone identifies an opportunity
1- **Elect Lead Squad** – where one squad becomes lead
2- **Analysis** – where the lead squad engage stakeholders to understand what's needed
3- **Engage Contributing Squads** – where the lead squad creates a relation to contributing squads
4- **Design and Plan the Solution** – where everyone design the solution together
5- **Manage the Delivery** – where the lead squad coordinates and contributing squads contribute
6- **Celebrate!** – where everyone has a party

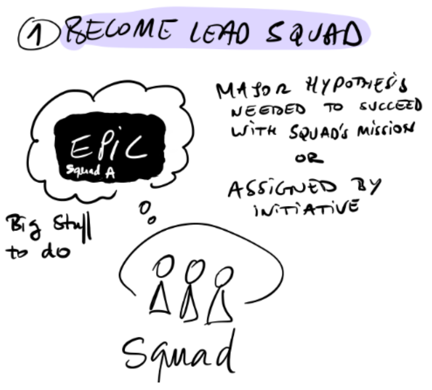
0- Discovery: An Opportunity has been Discovered!

Someone (your squad, some other squad, "the organization") has discovered an **opportunity**!

The opportunity can be formulated in different ways, mirroring the level of mandate coming with it. It could be an *Objective* to meet (maybe using the Objective and Key Result framework), a hypothesis to validate (a *Bet*), or a predefined function to implement (an *Epic*).

Christophe Achouiantz
christophe.achouiantz@crisp.se
Originally created for Telenor Sweden

1- Elect Lead Squad: A Squad Becomes Lead for some Development Work



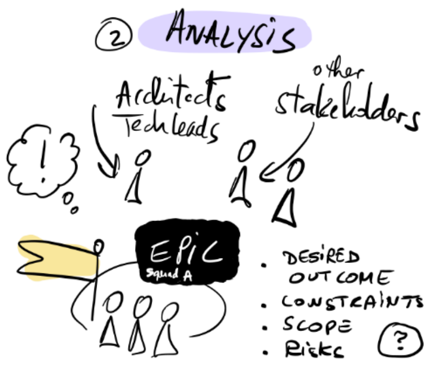
Congrats! Your squad will lead a development effort required to **seize an opportunity** (i.e. validate a business *hypothesis* / bet, or implement and *Epic*)! Several other squads may be needed to succeed with this opportunity, but you are **the Lead Squad**: the champion and coordinator for the effort.

As seen in phase 0, this development work may originate from your squad (i.e. it is directly related to your squad's mission), or your squad may have been *assigned* the work by your Tribe (somehow, your squad is best suited for helping the organization succeed with some important objective).

Who should be Lead Squad? Generally the Lead Squad is the squad that has the *most skin in the game*, either because the opportunity is directly related to the squad's mission, or because the squad will probably spend the most effort during development (while other squads - the contributing squads - will develop relatively smaller pieces). But this is not always true. In some case the Lead Squad has practically no development effort, but it is still an opportunity that is strongly related to their mission. In other cases, the most obvious squads could be too busy (perhaps leading other opportunities) so "another" squad with capacity was picked.

What do we mean by the squad is Lead Squad? Isn't this the PO's work to lead mission-related work? No, it is not an activity that only concerns the PO. Being Lead Squad means that **the squad as a whole takes responsibility** for driving the work. The PO and Agile Team Lead/Scrum Master will naturally be engaged to contribute on the product, alternative way-of-work, but it is not the rule that it must be this way. Perhaps your squad uses a "feature champion" or similar concept, in which case that person will take a lead role. Regardless, everyone should be involved, and different squads will have different setups.

2- Analysis: The Lead Squad Analyzes the Opportunity



The squad **engages relevant stakeholders** (tech leads, architects, business representants, privacy/security, etc.) to better understand what needs to be done.

The squad engages relevant stakeholders (tech leads, architects, business representants, privacy/security, etc.) to, together, analyze the opportunity and clarify:

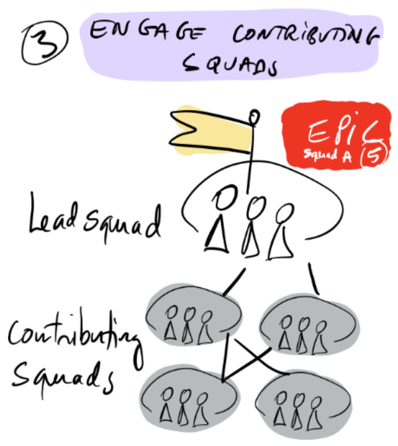
- **Why** - the Story. Why is this valuable? Why now? What is the impact if this opportunity is not pursued or fails? For whom? Who's the *customer* here?
- **Desired outcome** (formulated from a customer perspective) – usually enabling the customer to do new/different things, i.e. create or change customer behaviors.
- **Scope** (what must be in, what could be left out)
- **Constraints** (deadlines, specific knowledge needed, budget, resources, priority against other dev work, etc.) – everything that puts boundaries to the development effort.
- **Major risks**. More risks will be discovered as we go, for now try to surface the most obvious ones (strong dependencies, lack of knowledge/know-how, etc.) – the goal here is to be proactive.
- **Dependencies**: what other squads or stakeholders are needed to succeed with the development. This requires a high-level breakdown to understand what other squads (if any) may be needed to succeed with the development work. At this point, this should be a quick and dirty breakdown. Do not spend days trying to be "exact".

This is an initial high-level analysis and break-down of the opportunity, the point is not to go into all details (yet). For that you will need all the other contributing squads (later). So, at this point try to focus on the high-level and - most importantly - try to identify the squads you need to succeed.

Who, from the squad, is involved in this analysis? Ideally *all* team members. To get a stable solution, you want to engage as many members as possible in understanding the context (why, outcome, scope, etc.), so that everyone can contribute from the brainstorming all the way to the implementation and validation. If this is not possible (everyone is so busy!), then make sure that at least the senior members of the squad attend (PO, SM/ATL, UX, test, dev) and that you cover most aspects/roles.

When and how to do this analysis? For reasonably complex opportunities, you could use normal squad events like refinement or replenishment meetings to have the all squad attend. For more complex opportunities it's best to book separated workshops. Whole-day workshops may be needed.

3- Identify and Engage the Contributing Squads



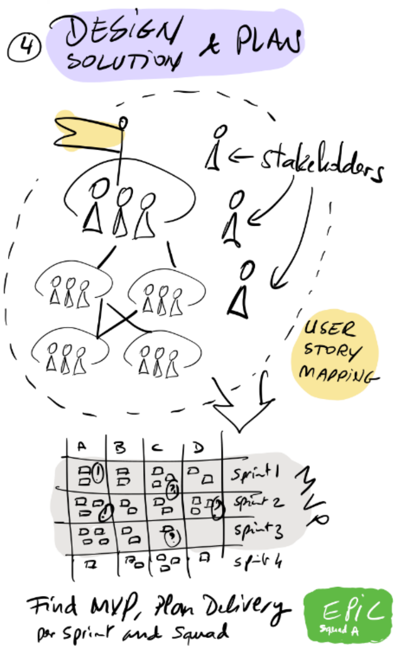
The initial high-level breakdown revealed dependencies on some squads. Those squads are the **contributing squads**. You may have also identified other **stakeholders** as well (e.g. security, privacy, non-development units in the rest of the organization).

You should now ***contact the contributing squads and stakeholders*** to engage them in the effort. Present the context (why, constraints, etc.) and together with them try to identify new risks / issues and yet other contributing squads and stakeholders.

Tips: Start already at this point to setup a meeting structure to share information (plans, issues/risks, activities, etc.) and to synch with the contributing squads and stakeholders. This meeting structure will be the backbone you need during phase 5. Start with a 30min weekly meeting and increase the cadence if/when needed. Also, start using some board (physical or digital depending on your context) to visualize the information.

Christophe Achouiantz
christophe.achouiantz@crisp.se
Originally created for Telenor Sweden

4- Design and Plan the Solution



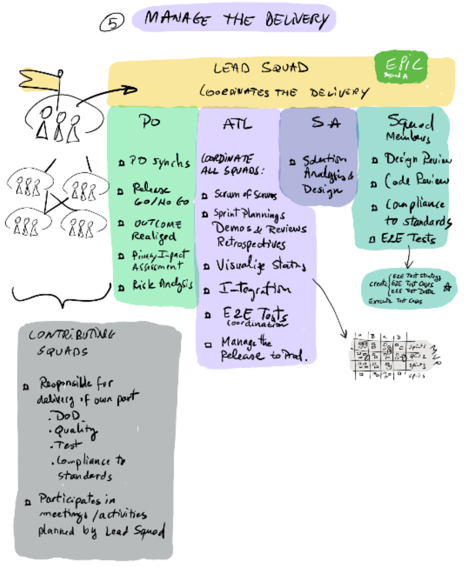
The Lead Squad involves all contributing squads and stakeholders in designing the solution together.

This is ideally done during a *workshop* (some hours to a couple a days) where you have representants from all the contributing squads and all the stakeholders. This can be done in different ways, but we strongly recommend using the **User Story Mapping** method. This helps you engage all participants in creating an **iteration plan** to produce an **MVP** (Minimal Viable Product). You can also decorate the map with issues, risks, questions that are raised during the discussions for later reference. The resulting map makes visible who does what, when and why. It should be continuously updated during the next phase to help manage the delivery.

More on the User Story Mapping method here:
https://www.jpattonassociates.com/user-story-mapping/

"Would a 'Big Room' User Story Mapping workshop work? Say, 10 squads, about 60+ people. It actually turned out quite OK when we tried that yesterday at Telenor. It created a lot of value by aligning all squads on what's most important, discussing what is possible and creating focus. We ended up with a solid user story map, outlining three iterations with clear goals (including end-to-end tests), and a pretty good list of questions and risks to address. The work will now be coordinated by the Lead Squad with weekly synchs at the product (PO), architecture (SA) and way-of-work (Scrum Master) level. Go Squads! ***#telenor #userstorymapping #bigroom"*** **– LinkedIn Post, Christophe Achouiantz, April 1 2019**

5- Manage the Delivery



Now is the "just do it" phase, the most intensive and long going. Your squad works tightly with other squad to develop and deliver what is needed to act on the opportunity. It requires continuous **coordination of efforts**, **handling of risks and issues**, and **getting feedback** on the product and the way-of-work.

Ideally you should use and update the **User Story Map** created earlier to visualize and manage the work (iterations, MVP, risks, questions, etc.). You could complement it with a **Kanban board** to show what is being worked on right now, by whom. This help you reflect on what can be done to improve flow of the work (tips: there are probably some queues building around Validation).

Responsibilities of the Lead Squad:

- Ensure that there is a clear **Definition of Done** for the development effort, so that you always know what's left to do when a squad report something as "done". Ideally, all squads should have the same DoD. If not, make it crystal clear and visible.
- **Product (POs), Architecture (SAs) and Engineering (dev, test) are always in synch**: POs synch on the priorities and risks, SAs on the break-down and design, Dev & Test on changes to common code base or results of latest tests. The Lead Squad organizes and hosts regular short "daily" meetings (at least weekly, but better twice per week, or even daily) for all three domains. The goal of the "daily" is to **share status, address risks and issues and plan the work** until next meeting. For small effort, a common "daily" for all domains works well. For bigger efforts, you may need to have three separated "dailies" (POs, SAs, Dev&Test) with different cadences to allow for deeper/longer discussions.
- **Visualization of the current effort, progress, dependencies, risks and plans**. The visualization should be accessible to all. Tips: use *User Story Map* and *Kanban boards*.
- Organizing the review of the contributing squads' design and code (if relevant).
- **Integration testing and validation** of the whole solution. The Lead Squad oversees the continuous integration of the contributing squads' effort.

Responsibilities of the contributing squads:

- Deliver their part of the development effort with **the expected quality**, **compliance to standards**, **respecting the agreed DoD**
- **Actively participate** in the coordination activities organized by the Lead Squad

Further recommendations to ensure a pro-active and efficient collaboration:

- **Go Make Contact!** Squads required to consume APIs from another's squad are expected to actively contact the squad, *by going and talking* to that squad. Squads that have periods of intensive collaboration (e.g. implementing API) are expected to sit together if possible or having longer video sessions (at least key persons).
- **Customer-centric!** Squads providing APIs to other squads are expected to take time to *onboard* these squads on how to use their API(s), as well as create time for supporting them when issues arise.
- **Customers First!** Squads should consider prioritizing the de-blocking of their “customer” squads over other work (e.g. problem-shooting, inform/educate, fix bugs, etc.).
- **Handshake Contracts!** Squads providing/consuming APIs are expected to create a contract between them that specifies what info is exchanged. The contract should be *explicit* and *agreed* by both parties. Changes to the contract must be communicated to the consuming squads (Go Take Contact!) and the version number must be changed.
- **Come take a fika/beer!** Personal contact is crucial to make these common effort work. To avoid building frustration, take some time to socialize in a more relax environment from time to time. It will pay off in the long term!

6- Celebrate!

The Lead Squad shall organize a **“Delivery Party”** when the development effort concludes. All contributing squads and Stakeholders shall be invited!

Usage at Telenor Sweden

Here are some lessons learned from applying the Lead Squad Protocol at Telenor Sweden 2018-2020.

Limitations

- As the coordination cost for the Lead Squad is directly proportional to the number of squads involved, there is a limit to the size and complexity of the opportunities that can be managed using the Lead Squad Protocol. We've managed to have one squad drive the work of 12 contributing squads for several months, but it was very taxing for the Lead Squad especially for the PO and ATL. The recommendation is now to **break down development effort** to have **a maximum of 5 contributing squads** per Epic.
- Being Lead Squad for large and complex efforts relies heavily on the people in key roles (PO, SM, SA) to take full responsibility for the Epic/Bet and be proactive. This requires a specific skill set, experience and personality, and may not be for "everyone".

Anti-patterns

- **The Scrum Master/ATL as project lead.** The more contributing squads and the bigger the development effort, the more time must be spent on coordination. Up to the point when the Scrum Master/ATL doesn't have the enough time to act as a SM/ATL for his/her squad, but only as a "project lead". This is not a viable solution. An emergency solution could be to engage someone to help/support the SM/ATL with the coordination. A longer-term solution is to break-down the development effort into smaller pieces to reach a situation with less complex efforts and with fewer contributing squads.
- **Leaders relying too much on the squads to handle all issues.** The Lead Squad Protocol forces squads to take a cross-squad responsibility (within a tribe or even cross-tribe) to tackle efforts larger than themselves. As the lead squads seem to take care of "everything", it may result in the leaders/managers not engaging sufficiently with the squads. We've had situations when risks, issues and problems were not addressed by the squads (due to a too hard focused on delivery) and not by the leaders/managers either as they trusted the Lead Squad to manage. The recommendation is to have leaders/managers even more present and engaged with the Lead Squads to ensure that cross-squads concerns are quickly addressed.
- **No supporting/service squads**. Coordinating many contributing squads will uncover many cross-squads/tribes concerns that will need to be addressed by the Lead Squad. A lot of effort and time can be saved by having dedicated support/service squads to address continuous integration and delivery, quality assurance frameworks (end-to-end testing, performance testing, etc.).